Institut für Parallele und Verteilte Systeme
Abteilung Bildverstehen

Universität Stuttgart
Universitätsstraße 38
70569 Stuttgart

Diplomarbeit Nr. 2792

Korrespondenzfindung zwischen Targetmengen im 3D-Raum

Ivan Bogicevic

Studiengang:	Softwaretechnik
Prüfer:	PD Dr. rer. nat. Michael Schanz
Betreuer:	Dr. Martin Ossig
	Dipl.-Inf. Alexander Kramer
	PD Dr. rer. nat. Michael Schanz

begonnen am:	01.07.2008
beendet am:	31.12.2008
CR-Klassifikation:	I.1.2, I.3.7, I.5

Inhaltsverzeichnis

Zusammenfassung / Abstract **7**

1. Einleitung **9**
1.1. Laserscanner 9
1.2. Registrierung 10
1.3. Aufbau 11

2. Anforderungen **13**
2.1. Rahmenbedingungen 13
2.2. Transformation 14
2.3. Targets 15
2.4. Einfache Fehlertoleranz 15

3. Lösungsansätze **17**
3.1. Konzept 17
3.2. Relationen 18
Einseitige Relationen 18
Dreiecksrelationen 19
Globale Relationen 20
Zweiseitige Relationen 20
3.3. Abstände und Winkel 21
Klassische Berechnung 21
Parallelabstand 22
3.4. Resultierende Fehlertoleranz 23
3.5. Hubs and Authorities 24
3.6. Precision and Recall 25

4. Algorithmus **27**
4.1. Übersicht 27
4.2. Berechnung der Relationen 28
4.3. Berechnung der Äquivalenzmengen 30
4.4. Initialisierung der Ähnlichkeitsmatrix 31
Aufbau der leeren Matrix 31
Setzen initialer Ähnlichkeitswerte 31
Setzen der Verbindungen 32
4.5. Iterierte Neuberechnung der Ähnlichkeitsmatrix 33
4.6. Bestimmung der Korrespondenzen 34
Bestimmung der Korrespondenzkandidaten 34
Wahl der besten Kandidaten 35

5. Varianten **37**
5.1. Gewichtung nach Abweichung 37
5.2. Eliminierung von Einträgen 38
5.3. Normierung der Matrix 38
5.4. Keine Testregistrierung 39
5.5. Ausschließliche Testregistrierung 39
5.6. Inkrementelle Neufindung 39

6. Analysen **41**
6.1. Parameter für Targetwolken 41
6.2. Metriken 43
6.3. Parameter zur Ähnlichkeitsmatrix 44
Variation von RECALC_NUM und LINE_MAX_THRESH 45
Variation von TOTAL_MAX_THRESH 46
6.4. Parameter zur Toleranz 47
Wahl von PARALLEL_DIST_MAX 48
Variation von PARALLEL_ANGLE_MAX 48
6.5. Robustheit 49

7. Theorie **51**
7.1. Definitionen 51
7.2. Korrektheit 52
Satz 52
Beweis 52
7.3. Komplexität 54
Voraussetzungen 54
Platzaufwand 54
Zeitaufwand 55
7.4. Matching Algorithmen 56
Graph Matching 56
Point Matching 57
7.5. Extrembeispiele 58
Mehrdeutige Lösungen 58
Wiederkehrende Muster 59

8. Implementierung **61**
8.1. Objekteigenschaften 61
8.2. Anwendungsfälle 62
Finde Korrespondenzen 62
Finde alle Korrespondenzen 63
Entfernen eines Objektes 63
Umbenennen eines Objektes 63
8.3. Korrespondenzfindung (Klasse CorrespondenceSearch) 64
8.4. Ähnlichkeitsmatrix (Klasse CorrespondenceTable) 66
8.5. Klassenübersicht 68
8.6. Korrespondenzprüfung 69

9. Markante Punkte **71**
9.1. Interest-Operatoren 71
9.2. FAST-Operator 72
9.3. Förstner-Operator 73
9.4. Optimierungen 74
Gleichverteilung 74
Musterpunkte 75
Identische Punkte 76
Kreuzungspunkte 76
Schattenlinien 77
Globale Selektion 77
9.5. Maskenskalierung 78
9.6. Kombination 79

10. Ausblick **81**

Literatur **83**

Verzeichnis **84**
Abbildungen 84
Tabellen 85
Diagramme 85
UML-Diagramme 85

Index **86**

Erklärung **92**

Zusammenfassung

Verfahren zur automatischen Registrierung von Daten aus 3D-Laserscannern basieren oft auf der Erkennung bestimmter Objekte im Scan, wie etwa Targets. Typischerweise wird dabei der Objekttyp identifizifert, nicht aber das Individuum. Dieser Schritt erfolgt mit Hilfe der Korrespondenzfindung, in der die geometrische Konstellation erkannt und zur individuellen Identifizierung verwendet wird.

In dieser Arbeit wird ein Verfahren vorgestellt, das eine solche automatische Korrespondenzfindung realisiert. Sowohl der Algorithmus selbst, als auch Teile seiner Analyse basieren auf einer Kombination aus bestehenden Graph Matching Algorithmen und Konzepten aus dem Information Retrieval.

Weiterhin berücksichtigt er variable Translations- und Rotationsfehler in den Scandaten und ist effizient, bei einer angenommenen Anzahl r von Targets des einen Scans beträgt der Zeitaufwand nur $O(r^2 \cdot \log(r))$, der Platzaufwand $O(r^2)$. Die hohe Geschwindigkeit und die Robustheit gegenüber Messfehlern und fehlenden oder zusätzlichen Targets wird anhand zahlreicher automatisierter und zufallsbasierter Tests bestätigt. Nach einer theoretischen Untersuchung wird eine mögliche Implementierung vorgestellt. Zusätzlich enthält diese Arbeit ein Kapitel zur automatischen Suche nach markanten Punkten im Raum, was als Vorstufe zur Korrespondenzfindung sehr nützlich ist, um z.B. Targets automatisch zu finden.

Abstract

Methods for the automatic registration of 3D laser scanner data are often based on the recognition of specific objects in space, like targets. Typically, the type of the object is thereby identified, but not the individual. This step is done by the search for correspondences, where geometric constellations are recognized and used for the individual identification.

In this thesis, a method is introduced, which realizes such an automatic search for correspondences. Both the algorithm itself and parts of its analysis are based on a combination of existing graph matching algorithms and concepts of information retrieval.

Furthermore, it considers variable errors in translation and rotation of scan data and is efficient, with an assumed number r of targets of one scan, the time effort is only $O(r^2 \cdot \log(r))$, the space effort $O(r^2)$. The high speed and the robustness due to errors in measurement and missing or spare targets is being confirmed by numerous automatic and random based tests. After a theoretic investigation, a possible implementation is presented. Additionally, this thesis contains a chapter about the automatic search of interesting points in space, which is useful as a pre-stage for the search for correspondences, e.g. to find targets automatically.

1. Einleitung

1.1. Laserscanner

Menschen erstellten mit Höhlenmalereien bereits zu prähistorischen Zeiten zweidimensionale Abbilder der dreidimensionalen Welt. Eine technische Möglichkeit, Bilder ohne kreatives Handeln zu erzeugen, gibt es seit Anfang des 19. Jahrhunderts mit der Fotografie. Dabei hat sich die Intention bis hin zur modernen Verwendung von Digitalkameras nicht wesentlich geändert. Es soll ein räumlicher und zeitlicher Ausschnitt der Realität festgehalten werden. Dabei ist meist der optische Eindruck wichtig, oft interessieren auch die Bemaßungen der festgehaltenen Objekte.

Ein Foto enthält jedoch kaum noch dreidimensionale Rauminformationen. Nach [SE07] ist ein Foto ein Modell seines Motivs. Dieses Modell enthält nicht mehr alle Attribute des Originals, was als Verkürzungsmerkmal zu sehen ist. Eine dieser Verkürzungen ist hier die Reduktionen von 3D auf 2D. Fotogrammetrische Methoden ermöglichen teilweise die Rekunstruktion dieser dritten Dimension.

Abbildung 1: 3D-Laserscanner, Quelle: [Faro08]

Bei 3D-Laserscannern sind diese Methoden jedoch nicht nötig. Ein rotierender Laserstrahl erfasst dabei berührungslos für jeden Raumwinkel die Entfernung zum nächsten Objekt, woraus Punktwolken entstehen, die am Rechner bearbeitet werden können. Hierfür gibt es zahlreiche Anwendungsbereiche. [Faro08] nennt folgende:

- Forensik: Tatortanalyse und Tatortdokumentation
- Architektur, Archäologie und Denkmalschutz
- Fabrik- und Anlagenbau
- Produktdesign: Gussprodukte, Betonbauteile, Flugzeuge
- Hoch- und Tiefbau: Tunnelbau, Höhlenvermessung

1.2. Registrierung

Um ein Gesamtbild eines von 3D-Laserscannern zu erfassenden Raumes zu erhalten, ist es im Normalfall nötig, den Raum von mehreren Standorten aus zu scannen. Registrierung ist das Zusammenführen von Scans unterschiedlicher Standorte zu einer Punktwolke im gemeinsamen Koordinatensystem.

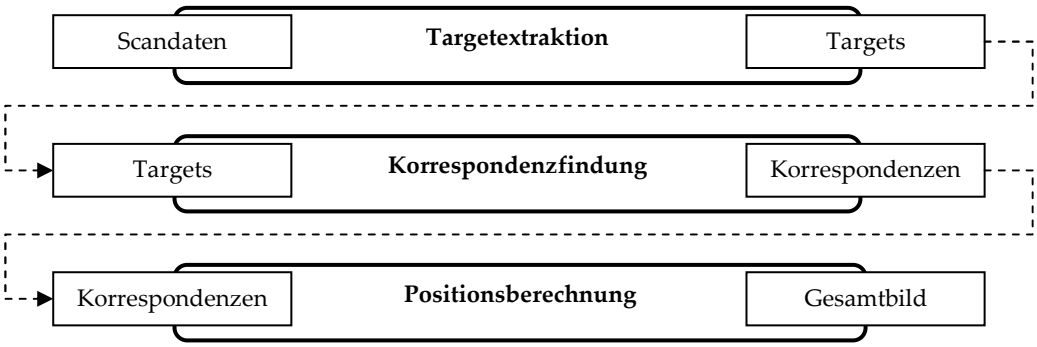


Abbildung 2: Registrierung – Ablauf

Folgendes Vorgehen ist möglich: Zunächst werden aus jedem Scan sogenannte Targets extrahiert. Targets sind 3D-Primitive wie Punkte, Kugeln, Ebenen und Zylinder. Nach der Erkennung dieser Targets aus den einzelnen Punktwolken ist der nächste wichtige Zwischenschritt die Korrespondenzfindung. Dabei wird festgestellt, welche Targets aus den verschiedenen Scans jeweils identisch sind, also im Raum demselben Objekt entsprechen. Diese Zuordnung wird Korrespondenz genannt.

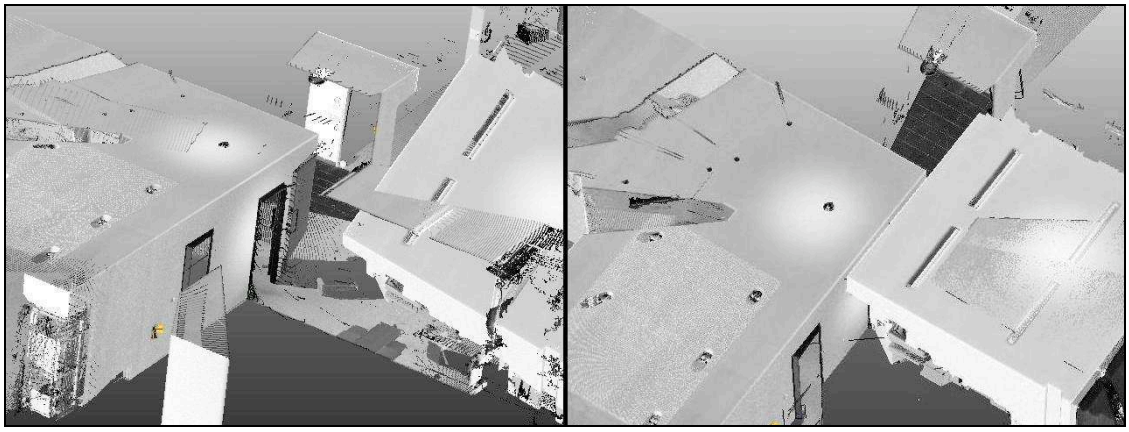


Abbildung 3: Registrierung – Beispiel

Sofern genügend zueinander identische Targets gefunden wurden, erfolgt die Berechnung der Positionen der einzelnen Scanbilder zueinander, woraus das Gesamtbild erzeugt werden kann. In dieser Arbeit wird ein Verfahren vorgestellt, welches eine automatische effiziente Korrespondenzfindung im 3D-Raum realisiert.

1.3. Aufbau

Diese Arbeit beginnt mit einer **Einleitung** (Kapitel 1) in die Thematik der Registrierung und Korrespondenzfindung. Danach werden verschiedene, sich aus der Aufgabenstellung ergebende, **Anforderungen** (Kapitel 2) beschrieben, die an den Algorithmus zur vollautomatischen Korrespondenzfindung gestellt werden. Insbesondere sind das der Rahmen der Ein- und Ausgabedaten, die Leistungen des Algorithmus selbst und die Arten der zu berücksichtigen Fehlertoleranzen.

Unter **Lösungsansätze** (Kapitel 3) wird ein gängiges Konzept vorgestellt, nach dem ein solcher Algorithmus arbeiten kann. Verschiedene selbst entwickelte mögliche Vorgehensweisen bei der Berechnung der Strukturen und ihrer geometrischen Maßzahlen in den einzelnen Scans werden betrachtet und bewertet. Die für den Algorithmus verwendeten und aus dem Information Retrieval kommenden Konzepte „Hubs and Authorities“ sowie „Precision and Recall“ werden vorgestellt.

Der Kern dieser Arbeit ist eine Beschreibung des entwickelten **Algorithmus** (Kapitel 4). Hier wird in einzelnen Schritten ausführlich erläutert, was das Verfahren jeweils macht. Zur Erleichterung des Verständnisses begleitet ein anschauliches Beispiel dieses Kapitel. Verschiedene selbst entworfene **Varianten** (Kapitel 5) des Algorithmus, die sich bei dessen Entwicklung angeboten haben, werden betrachtet und einzeln bewertet.

Im nächsten Kapitel sind die Ergebnisse der detaillierten **Analysen** (Kapitel 6), die im Rahmen dieser Arbeit mit dem Algorithmus durchgeführt wurden. Insbesondere werden die Testdaten und Metriken zur Bewertung der Ergebnisse beschrieben. Durch Variation verschiedener Parameter wird untersucht, wie der Einfluss auf das Ergebnis ist und welche Parameterkombination bei echten Scandaten die besten und robustesten Korrespondenzen liefern. Als nächstes wird die **Theorie** (Kapitel 7) des Verfahrens näher betrachtet. Nach einer Klassifikation werden bestimmte Aussagen über den Algorithmus getroffen und bewiesen. Eine Untersuchung des Zeit- und des Platzaufwandes findet statt. Im Vergleich dazu werden zwei bestehende Graph Matching Algorithmen vorgestellt. Zuletzt werden die Grenzen des Verfahrens durch Extrembeispiele erläutert.

Es folgt eine Darstellung der hier gemachten **Implementierung** (Kapitel 8) des Algorithmus, zunächst aus Sicht des Anwenders, wie die Benutzerschnittstelle und einzelne Anwendungsfälle aussehen. In mehreren UML-Diagrammen werden dann die einzelnen Aktivitäten und Klassen der Implementierung erläutert. Eine verwendete Methode zur Prüfung der Implementierung wird vorgestellt.

Zusätzlich folgen zwei bestehende Verfahren zur automatischen Suche **Markanter Punkte** (Kapitel 9) im Raum, ergänzt durch eigene und an echten Scandaten veranschaulichte Optimierungsmöglichkeiten dieser Operatoren.

Ein **Ausblick** (Kapitel 10) schließt die Arbeit ab.

2. Anforderungen

2.1. Rahmenbedingungen

In [Korr05] werden Faktoren genannt, die eine Klassifikation einer Registrierung und damit auch einer Korrespondenzfindung ermöglichen. Die folgende Einordnung ist eine Anwendung dieser Faktoren auf den Fall von Daten aus 3D-Laserscannern.

- Art der Bilddaten: Die Bilddaten sind Punktmengen im 3D-Raum.
- Bereich der Registrierung: Der von beiden Scans überdeckte Bereich ist nicht bekannt. Überhaupt keine oder eine komplette Überlappung ist auch möglich.
- Grad der Automatisierung: Die Korrespondenzfindung und die Registrierung soll nach der Targeterkennung komplett automatisch ablaufen. Insbesondere soll es nicht nötig sein, sich in den gescannten Räumen zu orientieren.
- Grad der Genauigkeit: Die Genauigkeit bei der Registrierung ist direkt abhängig von den Fehlern in den Positionen der gescannten Objekte.
- Art der Transformation: Die Überführung einer Punktwolke zur Einpassung in die andere entspricht einer Abbildung im 3D-Raum. Dabei handelt es sich nach [Math04] um eine starre und globale Transformation. Starr, weil es keine Verzerrungen oder Scherungen gibt, und global, weil die Transformation auf eine komplette Wolke angewendet wird.

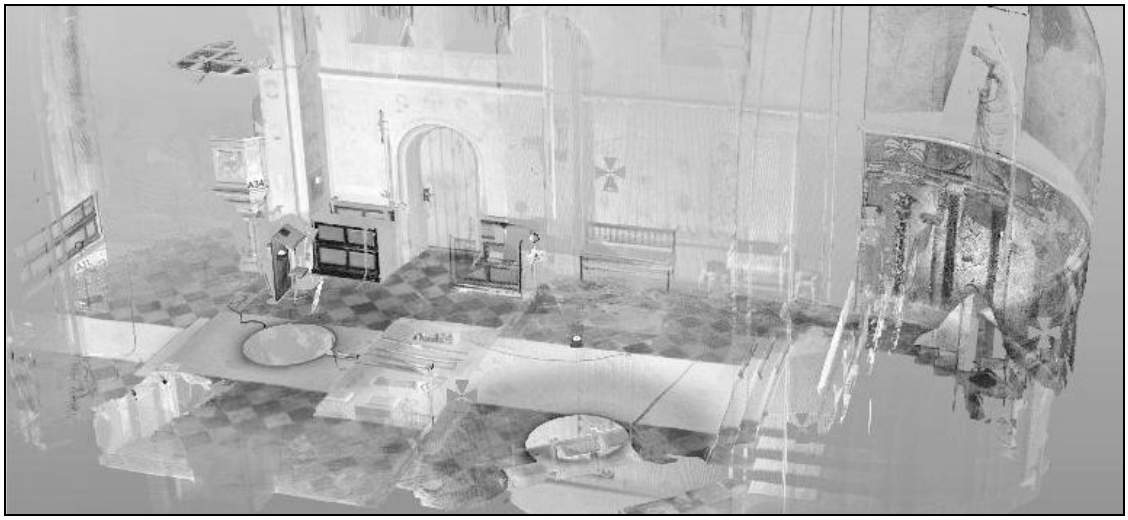


Abbildung 4: Dreidimensionalität der Bilddaten

2.2. Transformation

Die räumliche Lage eines Scanners kann durch sechs Parameter festgelegt werden. Drei Parameter x_1, y_1 und z_1 beschreiben die Position des Scanners, z.B. in kartesischen Koordinaten des R^3-Raumes. Drei weitere Parameter beschreiben die Drehung des Scanners um drei zueinander rechtwinklige Achsen, hierfür können z.B. aus der Luftfahrnorm [DIN9300] Gierwinkel Ψ, Nickwinkel Θ und Rollwinkel Φ verwendet werden.

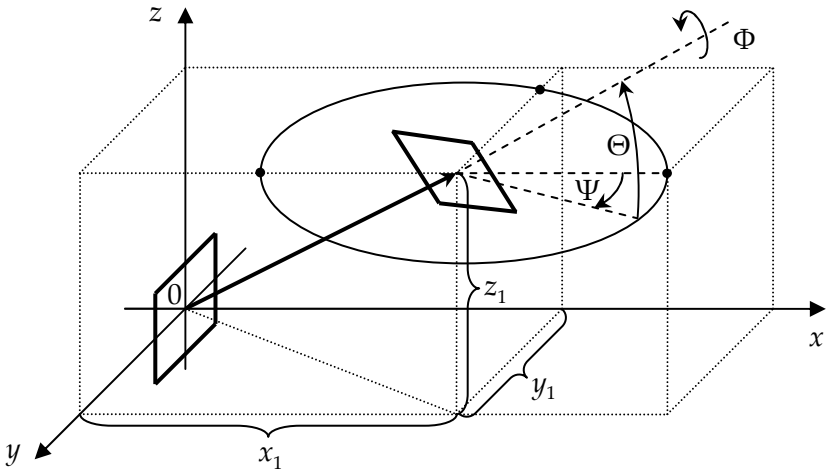


Abbildung 5: Parameter der Transformation

Ziel bei der Registrierung zweier Scans ist die Bestimmung dieser sechs Parameter. Der erste der beiden Scans, die Lokalwolke, wird dabei in den zweiten Scan, die Referenzwolke, eingepasst. Der Ursprung und die Richtungen der Koordinatenachsen der neu entstandenen gemeinsamen Wolke entsprechen denen der Referenzwolke. Diese neue Wolke ist nun die neue Referenzwolke. Jeder weitere hinzukommende Scan ist eine Lokalwolke, für die erneut die sechs Parameter berechnet werden müssen, um eine Registrierung zu ermöglichen.

Einige Scanner haben ein eingebautes Inklinometer, das mithilfe der Schwerkraft den Nickwinkel Θ und den Rollwinkel Φ bestimmen kann. Mithilfe eines Kompasses kann auch noch der Gierwinkel Ψ bestimmt werden, was jedoch durch magnetische Störfelder in den zu scannenden Räumen oft erschwert wird. Die Daten des Inklinometers und des Kompasses können die Anzahl der für eine eindeutige Registrierung notwendigen Targets deutlich reduzieren. Der hier vorgestellte Algorithmus funktioniert jedoch auch ohne diese Daten, da einige Scanner weder Inklinometer noch Kompass besitzen.

2.3. Targets

Natürliche Targets kommen ohnehin im Scanbild vor, z.B. eine als Ebene erkannte Raumwand. Spezielle Targets sind absichtlich im Scanbereich angebracht, z.B. auf Papier ausgedruckte Muster oder in den Scanbereich gelegte Kugeln. Jedes Target kann abhängig von seiner Art in eine der Grundtypen Punkt, Linie oder Ebene eingeordnet werden. Der in dieser Arbeit vorgestellte Algorithmus zur Korrespondenzfindung berücksichtigt die in Tabelle 1 aufgeführten Targets, wobei eine Erweiterung nur geringen Änderungsaufwand verursachen würde.

Grundtyp	Punkt	Gerade	Ebene
Untertypen	Kugel, Kreis, Schachbrettmuster	Zylinder	Wand, ebene Fläche
Lage	Positionsvektor $\vec{p}$	Positionsvektor $\vec{p}$, Richtungsvektor $\vec{u}$	Positionsvektor $\vec{p}$, Normalenvektor $\vec{n}$
Eigenschaften	Name, Wolke, evtl. Kugelradius	Name, Wolke, evtl. Zylinderradius	Name, Wolke, evtl. Wanddicke

Tabelle 1: Aufbau von Targets

2.4. Einfache Fehlertoleranz

Scandaten sind Messdaten und unterliegen demzufolge einer gerätespezifischen Messunsicherheit. Daher gibt es immer Abweichungen zur Realität, auch wenn diese teilweise nur sehr klein sind. Eine Anforderung, die der Algorithmus erfüllen muss, ist eine gewisse Toleranz gegenüber diesen Fehlern. Abhängig davon, um welche Targets es sich handelt und wie ihre relative Lage zueinander ist, unterscheidet sich die resultierende Toleranz. Am Anfang werden zwei Parameter festgelegt:

- $t_{err}[LE]$ ist der maximale Translationsfehler in Längeneinheiten. Er beschreibt, wie weit die gescannten Positionen der Targets von denen in der Realität maximal abweichen.
- $r_{err}[°]$ ist der maximale Rotationsfehler in Grad. Er beschreibt, um welchen Winkel die Richtungs- und Normalenvektoren der Targets aus dem Scanbild mit denen in der Realität maximal abweichen.

Bei diesen Parametern ist nicht nur die Messgenauigkeit des Scanners zu berücksichtigen, sondern auch das Maß, wie stationär die Positionen der Targets sind. Je geringer sich diese von Scan zu Scan unterscheiden, desto stationärer sind sie.

3. Lösungsansätze

3.1. Konzept

Ein naheliegendes Konzept zur Realisation der Korrespondenzfindung ist es, aus allen Targets eines Scans bestimmte Eigenschaften zu extrahieren, die bezüglich Translation und Rotation invariant sind. Eine invariante Eigenschaft ist z.B. der Abstand zweier Targets. Auf diese Vorgehensweise basieren viele Verfahren, wie z.B. auch die in [Corr86] und [Korr05].

Im Zuge dieser Arbeit hier wurden viele weitere solcher Attribute ausgedacht und untersucht, wobei die meisten wieder verworfen wurden, wie in den folgenden Unterabschnitten beschrieben. Die besten dieser Konzepte wurden dann für den Algorithmus in Kapitel 4 übernommen.

Aus den gewählten invarianten Eigenschaften wird für jeden einzelnen Scan eine Datenstruktur erstellt, z.B. ein Graph pro Scan, dessen Knoten die Targets und dessen Kanten die Abstände der Targets zueinander darstellen. Muster, die in Datenstrukturen unterschiedlicher Scans vorkommen und ähnlich zueinander sind, werden als korrespondierend erkannt. Daraus lassen sich dann unter bestimmten Voraussetzungen die Korrespondenzen der Targets ermitteln.

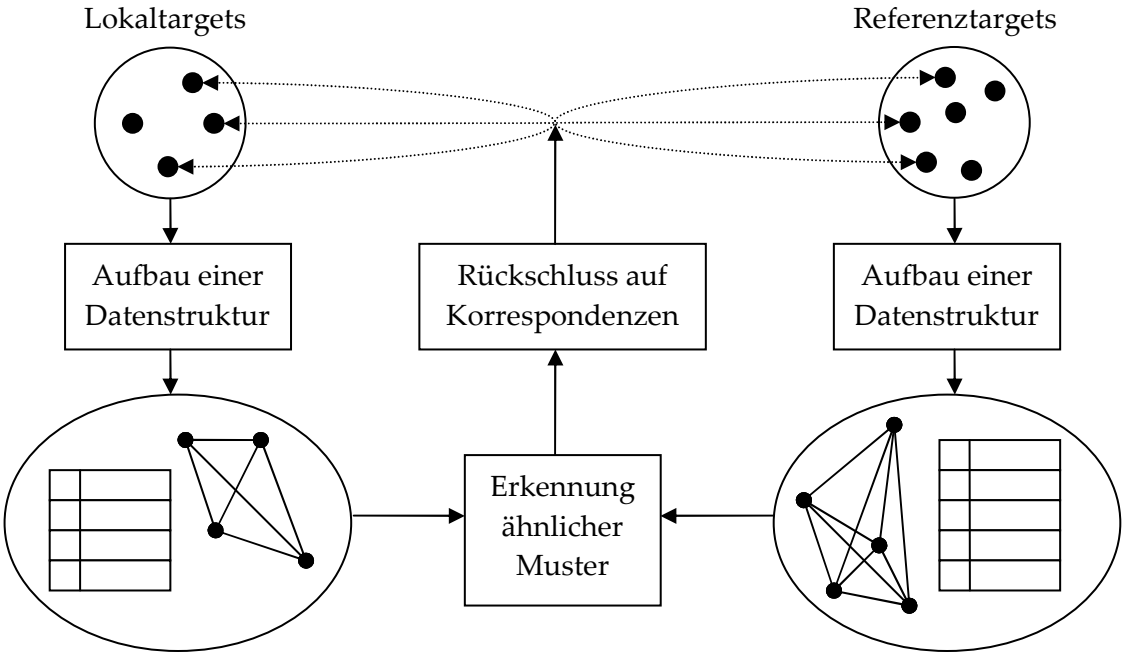


Abbildung 6: Ablauf einer Korrespondenzfindung

3.2. Relationen

Um die Datenstrukturen aufzubauen, können Targets einer Wolke miteinander in Beziehung gebracht werden, woraus Relationen entstehen. Relationen sind invariante Eigenschaften von Targets, die sich bei einer Translation und Rotation einer Wolke nicht ändern. Oft können mit ihnen die relativen Positionen der Targets im Raum beschreiben werden. Abhängig davon, wie viele Targets bei der Bildung einer Relation mitwirken, können Relationen in verschiedene Kategorien eingeteilt werden.

Einseitige Relationen

Relationen, die aus folgenden Eigenschaften generiert wurden, bestehen aus genau einem Target:

- Lageunabhängige Eigenschaften:
  - Grundtyp des Targets: Punkt, Gerade oder Ebene.
  - Untertyp des Targets: Eine Kugel und ein Kreis sind zwar beide Punkttargets, über den Untertyp können diese jedoch unterschieden werden.
  - Radius: Zwei Zylinder oder zwei Kugeln haben zwar jeweils den gleichen Untertyp, bei unterschiedlichen Radien ist aber dennoch eine eindeutige Zuordnung möglich.
- Lageabhängige Eigenschaften:
  - Aus den Positionsvektoren der Targets lassen sich keine sinnvollen Werte ableiten, die bezüglich beliebigen Translationen und Rotationen invariant sind. Denkbar wäre der Betrag des Vektors (Abstand zum Ursprung), dieser ist jedoch nur gegenüber der Rotation invariant.
  - Bei den Richtungsvektoren von Geraden- und Ebenentargets ist die Situation ähnlich. Durch die Vektorenschreibweise ist zwar jede Translation ausgeglichen, eine mögliche Rotation jedoch nicht.

Fazit:

Die lageabhängigen Eigenschaften der Targets kommen hier nicht zur Geltung, obwohl diese eine große Aussagekraft besitzen. Wenn es viele Targets desselben Untertyps gibt, entstehen zu viele Mehrdeutigkeiten. Deshalb reicht die alleinige Verwendung von Relationen aus einem Target hier bei weitem nicht aus.

Dreiecksrelationen

Bei diesen Relationen werden immer drei Targets zueinander in Beziehung gebracht. In vielen Fällen entsteht dabei ein Dreieck aus

- drei Punkttargets oder
- zwei Punkttargets und einem Geradentarget oder
- zwei Punkttargets und einem Ebenentarget.

Aus diesem Dreieck können bestimmte Eigenschaften, wie etwa Winkel oder Streckenverhältnisse, berechnet werden, die in anderen Scans mit denselben Targets wiederkehren. Der wesentliche Nachteil dieser Methode liegt im nötigen Aufwand. Aus n Targets entstehen $\binom{n}{3} = \frac{n!}{3!(n-3)!} = \frac{n(n-1)(n-2)}{6} = O(n^3)$ viele Dreiecke.

Es ist nötig, alle Dreieckskombinationen der Targets zu berücksichtigen. Sobald Kombinationen von Targets ausgelassen werden, wie in Abbildung 7, können in bestimmten Situationen ähnliche Dreiecke nicht mehr zugeordnet werden, sogar wenn die zwei Wolken deckungsgleich sind. Für Relationen mit zwei Targets ist diese Problematik zwar auch vorhanden, dort fällt sie aber nicht so sehr in Gewicht, da es weniger Relationen pro Target gibt. Erschwerend kommt in beiden Fällen hinzu, dass man nicht weiß, welche Targets in einem Scan vorkommen, die in keinem anderen Scan vorhanden sind.

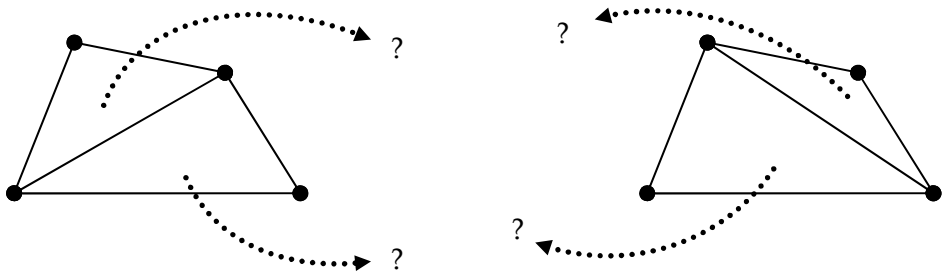


Abbildung 7: Ähnlichkeitssuche bei fehlenden Targetkombinationen

Fazit:

Der Aufwand zur Erzeugung der Relationen ist relativ hoch. Vorteile bei dieser Lösung sind nicht erkennbar.

Globale Relationen

Eine weitere Idee ist es, alle Targets eines Scans in die Berechnung mit einzubeziehen. In einer Wolke von Punkttargets könnte z.B. der Schwerpunkt oder auch die lineare Hülle berechnet werden. Weiß man aus anderen Gründen, dass in einem anderen Scan die Targets dieselben sind, es also keine Targets gibt, die nur in einem Scan vorkommen, dann wäre wenigstens die Translation berechenbar. In diesem Fall müsste man nur die eine Wolke soweit verschieben, dass ihr Schwerpunkt mit dem der anderen Wolke übereinstimmt. Die dabei gemachten Annahmen sind aber in den seltensten Fällen zutreffend, es gibt fast immer Targets, die nicht in allen Targetwolken vorkommen.

Versucht man, ähnliche Polygone mit mehr als drei Targets in den Wolken wiederzufinden, ergeben sich die gleichen Probleme wie bei Relationen mit drei Targets.

Fazit:

Die Anforderungen an die Targetwolken sind hier zu hoch, weshalb die Aussagekraft einer solchen Relation nur sehr gering wäre. Vorteile sind nicht erkennbar.

Zweiseitige Relationen

Hier besteht eine Relation aus zwei Targets. Aus n Targets entstehen $\binom{n}{2} = \frac{n(n-1)}{2} = O(n^2)$ viele Relationen.

Eine Relation verbindet jeweils zwei Targets einer Wolke. Sie kann als Kante eines Graphen verstanden werden, der aus den Targetwolken generiert wurde. Eine Relation aus zwei Targets hat zunächst einmal eine von sechs möglichen Relationstypen, abhängig von den drei Grundtypen Punkt, Gerade und Ebene ihrer beiden Targets T_1 und T_2. Der Skalar s einer Relation beschreibt mit seinem Wert die Relation. Hierfür kommen prinzipiell Abstände und Winkel zwischen den Targets in Frage, ausführlich erläutert wird das im folgenden Kapitel. Vektoren, wie z.B. der Verbindungsvektor zwischen zwei Punkttargets sind nicht direkt verwendbar, da sie sich bei einer Rotation ändern.

Fazit:

Es handelt sich um eine praktikable Lösung mit überschaubarem Aufwand zur Erzeugung der Relationen. Werden zusätzlich die Eigenschaften von Relationen aus einem Target mit berücksichtigt, ergeben sich sehr gut verwendbare Datenstrukturen.

3.3. Abstände und Winkel

Klassische Berechnung

Bei zweiseitigen Relationen mit mindestens einem Punkttarget ist die Verwendung von Abständen als skalare Beschreibung problemlos möglich, indem die klassische Berechnung der Distanz zweier Objekte genutzt wird. Winkel sind hier nicht geeignet, da bei Punkt–Punkt Relationen es ohne einen dritten Punkt gar keinen Winkel gibt. Bei Punkt–Gerade und bei Punkt–Ebene Relationen könnte als dritter Punkt der Stützpunkt der Gerade bzw. Ebene verwendet werden. Gegen eine solche Vorgehensweise spricht, dass dieser nicht eindeutig ist, und sich somit für dasselbe Objekt von Scan zu Scan unterscheiden kann.

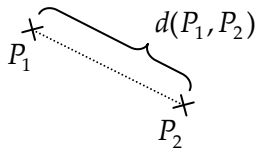
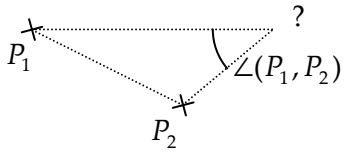
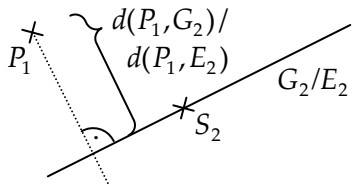
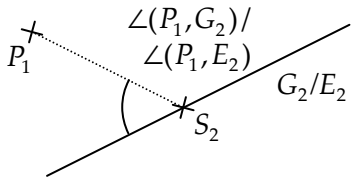
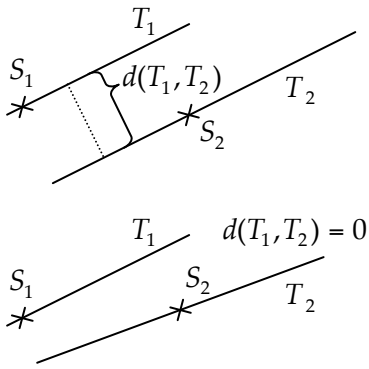
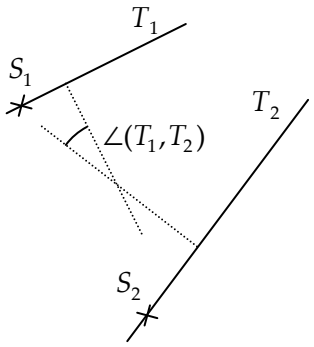
Relationstyp	**Abstand**	**Winkel**
Punkt–Punkt	problemlos möglich  ✓	nicht möglich  ✗
Punkt–Gerade Punkt–Ebene	problemlos möglich  ✓	nur mit Stützpunkt S_2 möglich  ✗
Gerade–Gerade Gerade–Ebene Ebene–Ebene	problematisch, wenn $T_1 \nparallel T_2$  ✗	problemlos möglich  ✓

Tabelle 2: Abstände und Winkel zwischen Targets

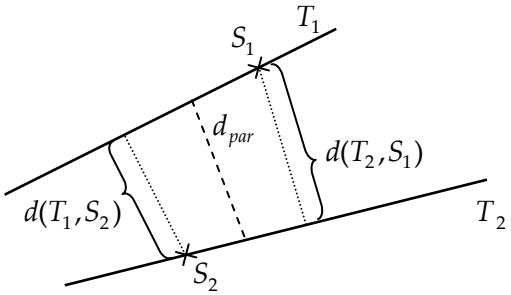
Parallelabstand

Bei Relationen ohne Punkttargets ist es aus mehreren Gründen problematisch, den klassischen Abstand als skalare Beschreibung zu verwenden:

- Der Abstand hat eine geringe Aussagekraft, da dieser immer 0 ist, wenn sich die beiden Targets schneiden. Bei Ebene–Ebene und Ebene–Gerade Relationen ist das immer dann der Fall, wenn die Targets nicht parallel sind oder ineinander liegen, was sehr oft zutrifft.
- Der Abstand springt plötzlich von einem beliebigen Wert auf 0, sobald sich eines von zwei zueinander parallelen und nicht ineinander liegenden Targets minimal dreht und einen Schnitt verursacht.

Eine alternative Definition des Abstandes zweier Objekte ist es, den Abstand zwischen ihren Stützpunkten zu verwenden. Das ist jedoch keine gute Wahl, denn diese Positionen sind nicht eindeutig, wodurch sich der Abstand erheblich unterscheiden kann. Selbst wenn man z.B. immer den Schwerpunkt des sichtbaren Ebenen- oder Geradenausschnittes als Stützpunkt wählt, kann sich immer noch dieser Ausschnitt von Scan zu Scan und damit auch der Schwerpunkt sehr unterscheiden.

Die Lösung ist eine Erweiterung des Abstandbegriffes auf Geraden und Ebenen, welche wegen eines Schnittes einen klassischen Abstand von 0 haben, jedoch innerhalb bestimmter Grenzen als ungefähr parallel ausgerichtet gelten. Der hier sogenannte Parallelabstand ist der Durchschnitt aus den Abständen zwischen einem Target und dem Stützpunkt des anderen Targets und umgekehrt.



$$d_{par} = \frac{d(T_1,S_2) + d(T_2,S_1)}{2}$$

Abbildung 8: Parallelabstand

Die Vorteile dieser Definition sind:

- Der Parallelabstand springt nicht bei kontinuierlichen Drehungen der Objekte.
- Bei parallelen Objekten ist der Wert mit dem klassischen Abstand identisch.
- Verschiedene Positionierungen der Stützpunkte wirken sich nur gering aus.

3.4. Resultierende Fehlertoleranz

Für jede Relation wird eine resultierende Toleranz berechnet, die sich aus den einfachen Toleranzen t_{err}, r_{err} und den Typen und Positionen der Targets der Relation zusammensetzt. Diese Fehlertoleranz drückt die maximal akzeptierte Abweichung des Skalars der Relation aus.

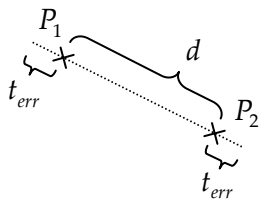
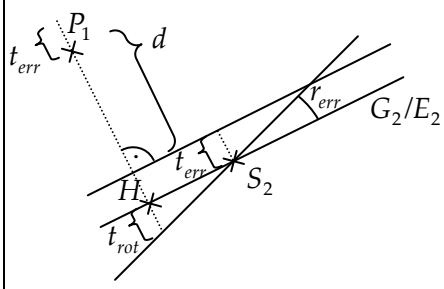
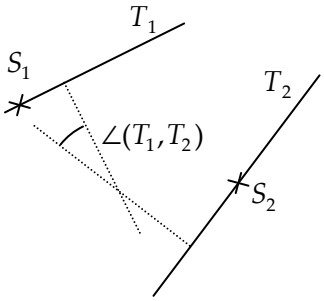
Relationstyp	**Herleitung**
Punkt–Punkt	 Im schlechtesten Fall bewegt sich jeder der beiden Punkte in entgegengesetzter Richtung um t_{err} voneinander weg. Der maximale Fehler im Skalar dieser Relation $d(P_1, P_2)$ ist somit $\delta_{PP} = 2t_{err}$ $[LE]$.
Punkt–Gerade Punkt–Ebene	 Im schlechtesten Fall bewegt sich das Punkttarget P_1 und der Positionspunkt S_2 der Gerade bzw. Ebene um t_{err} voneinander weg. Zusätzlich rotiert die Gerade bzw. Ebene um den Winkel r_{err} in die Richtung, in der der durch die Rotation entstandene Translationsfehler t_{rot} zu den beiden anderen Fehlern hinzukommt. Der maximale Fehler ist somit $\delta_{PG/PE} = 2t_{err} + t_{rot}$, wobei gilt: $t_{rot} = \tan r_{err} \cdot d(S_2, H) = \tan r_{err} \cdot \sqrt{d(P_1, S_2)^2 - d(P_1, T_2)^2}$. Die durch die Rotation verursachte Verschiebung des Hilfspunktes H kann wegen des geringen Winkels vernachlässigt werden. Der maximale Fehler im Skalar $d(P_1, G_2/E_2)$ ist also: $\delta_{PG/PE} = 2t_{err} + \tan r_{err} \sqrt{d(P_1, S_2)^2 - d(P_1, T_2)^2}$ $[LE]$
Gerade–Gerade Gerade–Ebene Ebene–Ebene	 Im schlechtesten Fall rotiert jedes Target um den Winkel r_{err} in der Richtung, in der der Winkel zwischen den beiden Targets sich jeweils um r_{err} vergrößert. Der maximale Fehler im Skalar dieser Relation $\angle(T_1, T_2)$ ist also $\delta_{GG/GE/EE} = 2r_{err}$ [°].

Tabelle 3: Herleitung der tolerierbaren Fehler

3.5. Hubs and Authorities

Um effizient aussagekräftige Informationen aus vernetzten Strukturen zu extrahieren, gibt es die Methode der „Hubs and Authorities“, die in [AuSrc98] beschrieben wird. Das Verfahren hat ein ähnliches Konzept wie das von Google verwendete PageRank [Page99] zur Bewertung von Webseiten im Internet. Eine Beschreibung beider Verfahren mit anschaulichen Beispielen befindet sich in [IR08].

„Hubs and Authorities“ weist jeder Webseite einen Hub-Wert und einen Authority-Wert zu. Einen hohen Authority-Wert haben typischerweise Seiten, die die eigentlichen Informationen und Inhalte besitzen, wie z.B. Webseiten großer Firmen oder Universitäten.

Wenn eine Webseite auf Seiten verlinkt, die einen hohen Authority-Wert haben, dann erhält sie einen hohen Hub-Wert. Typischerweise sind das Seiten, die gute Listen von Links besitzen. Umgekehrt errechnet sich der Authority-Wert einer Seite aus den Hub-Werten der Seiten, die auf sie zeigen.

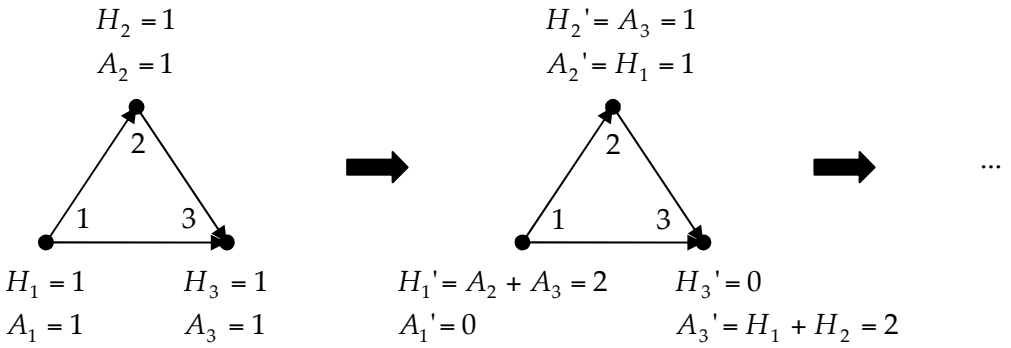


Abbildung 9: Hubs- and Authorities Berechnung

Der Grundsatz dieser Methode der wechselseitigen Berechnung der Werte wird in dieser Arbeit für den Algorithmus zur Korrespondenzfindung verwendet. Statt Webseiten, die mit Links verbunden sind, gibt es hier Wolken von Targets, die durch die in Abschnitt 3.2 beschriebenen Relationen verbunden sind. Es findet, ähnlich wie bei PageRank, keine Unterscheidung mehr zwischen Hub und Authority statt. Die rekursive Definition des Wertes einer Webseite, welcher dann hoch ist, wenn seine Nachbarseiten einen hohen Wert haben, wird hier auf die Korrespondenzfindung übertragen. Zwei Targets aus verschiedenen Wolken gelten als ähnlich, wenn ihre Nachbartargets als ähnlich zueinander gelten. Hierbei werden nicht alle Nachbartargets verwendet, sondern nur diejenigen, zu denen als gleich geltende Relationen führen. Dieser relativ komplexe Mechanismus findet schnell in sehr wenigen Schritten viele richtige Korrespondenzen. Im Kapitel zum Algorithmus wird er mit anschaulichem Beispiel ausführlich erläutert.

3.6. Precision and Recall

In [IR08] ist beschrieben, wie im Information Retrieval der Erfolg einer Anfrage mit bestimmten Metriken bewertet werden kann. Werden z.B. bestimmte Begriffe in eine Internet-Suchmaschine eingegeben und die gefundenen Ergebnisse anzeigt, so wird zwischen folgenden Mengen unterschieden:

- Gefundene Elemente: Alle von der Suchmaschine zu dieser Anfrage gefundenen Ergebnisse.
- Relevante Elemente: Alle Einträge der Suchmaschine, die der Benutzer mit seiner Anfrage tatsächlich finden möchte.

Daraus können die Größen Precision P und Recall R abgeleitet werden:

$$P = \frac{\text{\# relevante gefundene Elemente}}{\text{\# gefundene Elemente}}, \; R = \frac{\text{\# relevante gefundene Elemente}}{\text{\# relevante Elemente}}$$

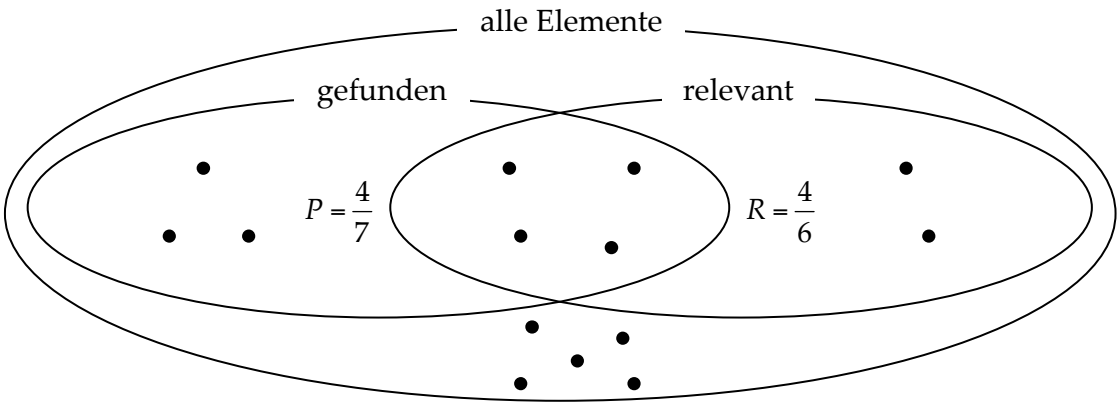


Abbildung 10: Beispiel zu Precision und Recall

Diese beiden positiven Metriken, im Idealfall sind beide 1.0, können mit einigen Änderungen auf die Korrespondenzfindung übertragen werden. Dabei gelten folgende Entsprechungen:

- Gefundene Elemente: Alle vom Algorithmus gefundenen Korrespondenzen.
- Relevante Elemente: Alle korrekten Korrespondenzen.

Aus der Metrik P folgen direkt die Metriken k und i. Dabei ist k der Anteil an korrekten und i der Anteil an inkorrekten Korrespondenzen, die der Algorithmus gefunden hat. Im Idealfall ist $k = 1.0$ (positive Metrik) und $i = 0.0$ (negative Metrik). Eine genaue Definition mit Beispiel befindet sich im Abschnitt 6.2: Metriken

4. Algorithmus

4.1. Übersicht

Der komplett in dieser Arbeit entwickelte Algorithmus realisiert eine automatische Korrespondenzfindung von Targetmengen im 3D-Raum. Er basiert auf den Vorbereitungen der beiden vorigen Kapitel, wo bereits Begründungen für einzelne Schritte stattgefunden haben. Das Verfahren besteht aus fünf aufeinander folgenden Schritten, die in den nächsten Unterabschnitten ausführlich erläutert werden:

1. Berechnung der Relationen
2. Berechnung der Äquivalenzmengen
3. Initialisierung der Ähnlichkeitsmatrix
4. Iterierte Neuberechnung der Ähnlichkeitsmatrix
5. Bestimmung der Korrespondenzen

Als Eingabe erhält der Algorithmus zwei Mengen von Targets, die Lokalmenge und die Referenzmenge. Der Aufbau der Targets selbst wurde bereits beschrieben. Die Ausgabe des Algorithmus sind alle gefundenen Korrespondenzen.

Beispiel

Am Ende jedes Unterabschnittes, in dem einer der Schritte des Algorithmus erklärt wird, befindet sich ein fortlaufendes Beispiel. Die Eingabe für den Algorithmus besteht bei dieser Beispielrechnung aus einer Lokalwolke mit den Punkttargets A, B, C und D und aus einer Referenzwolke mit den Punkttargets E, F, G und H, wie in Abbildung 11 dargestellt. Hier im einfachen Beispiel werden nur Punkttargets verwendet, mit anderen Targets funktioniert das Verfahren jedoch analog.

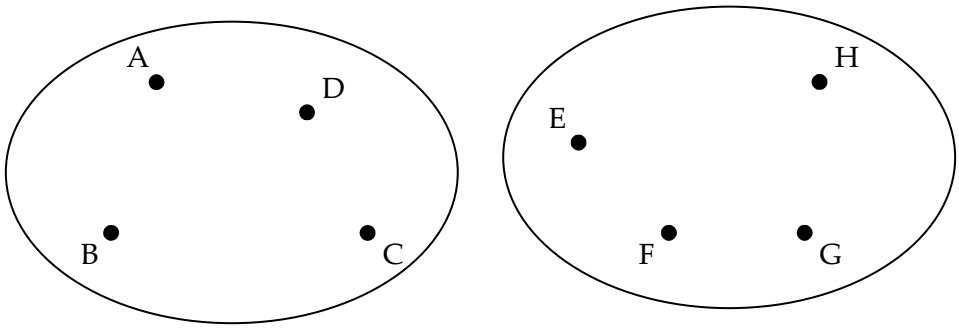


Abbildung 11: Lokal- und Referenzwolke als Eingabe

4.2. Berechnung der Relationen

Zuerst werden sowohl von allen Targets der Lokalmenge als auch von allen Targets der Referenzmenge die Relationen aufgebaut, wobei eine Relation immer zwei Targets derselben Wolke miteinander in Beziehung bringt. Aus den bisherigen Vorbereitungen ergibt sich folgendes Vorgehen: Ist eines der Targets der Relation ein Punkt, dann ist der beschreibende Skalar der Abstand von diesem Punkt zum anderen Target. Kommt in der Relation kein Punkt vor, dann ist der Skalar der Winkel zwischen den beiden Lagevektoren der Targets. Bei Geraden ist das der Richtungsvektor, bei Ebenen der Normalenvektor. Zu jeder Relation gibt es eine resultierende Toleranz, die festlegt, wie groß die maximale Abweichung des Skalars sein darf. Die Herleitung dazu befindet sich in Tabelle 3.

Relationstyp	**Skalar s**	**Resultierende Toleranz δ**	**Parallelabstand** d_{par}
Punkt–Punkt	$s_{PP} = d(P_1, P_2)$	$\delta_{PP} = 2t_{err}\ [LE]$	-
Punkt–Gerade	$s_{PG} = d(P_1, G_2)$	$\delta_{PG} = 2t_{err} + t_{rot}\ [LE]$	-
Punkt–Ebene	$s_{PE} = d(P_1, E_2)$	$\delta_{PE} = 2t_{err} + t_{rot}\ [LE]$	-
Gerade–Gerade	$s_{GG} = \angle(\vec{u}_1, \vec{u}_2)$	$\delta_{GG} = 2r_{err}\ [°]$	$d_{GG} = \frac{d(G_1, S_2) + d(G_2, S_1)}{2}$
Gerade–Ebene	$s_{GE} = \angle(\vec{u}_1, \vec{n}_2)$	$\delta_{GE} = 2r_{err}\ [°]$	$d_{GE} = \frac{d(G_1, S_2) + d(E_2, S_1)}{2}$
Ebene–Ebene	$s_{EE} = \angle(\vec{n}_1, \vec{n}_2)$	$\delta_{EE} = 2r_{err}\ [°]$	$d_{GG} = \frac{d(E_1, S_2) + d(E_2, S_1)}{2}$

Tabelle 4: Aufbau von Relationen

Zu jeder Relation gehört also die Information, auf welche beiden Targets sie sich bezieht, wie ihr skalarer Wert ist und wie groß der Wert der maximal akzeptablen (resultierenden) Toleranz ist. Zuzüglich wird der Parallelabstand der beiden Targets berechnet und gespeichert, falls die Targets innerhalb der Parameter als parallel gelten. Um den späteren Aufwand beim Durchgehen der Listen zu minimieren, wird hier sowohl die Liste der Relationen der Lokalmenge als auch die der Referenzmenge jeweils aufsteigend nach ihrem skalaren Wert sortiert.

Beispiel

Es entstehen folgende zwei ungerichtete gewichtete Graphen, wobei die Knoten den Targets entsprechen, die Kanten den Relationen und die Gewichte an den Kanten den einzelnen skalaren Werten. Da hier im Beispiel nur Punkttargets vorhanden sind, bezeichnen alle Werte an den Kanten hier die Abstände der jeweiligen Targets zueinander. Hier wird ein maximaler Translationsfehler von $t_{err} = 0{,}1$ angenommen, was zu einer resultierenden Toleranz von 0,2 führt.

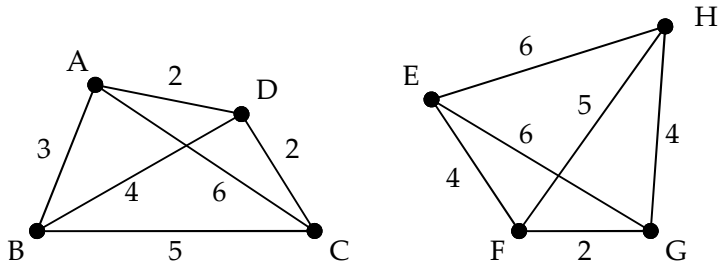


Abbildung 12: Relationen als Kanten eines Graphen

Lokal-relation	**Skalar**	**Toleranz**	**Referenz-relation**	**Skalar**	**Toleranz**
AB	3	0,2	EF	4	0,2
AC	6	0,2	EG	6	0,2
AD	2	0,2	EH	6	0,2
BC	5	0,2	FG	2	0,2
BD	4	0,2	FH	5	0,2
CD	2	0,2	GH	4	0,2

Tabelle 5: Relationen als Listeneinträge

Lokal-relation	**Skalar**	**Toleranz**	**Referenz-relation**	**Skalar**	**Toleranz**
AD	2	0,2	FG	2	0,2
CD	2	0,2	EF	4	0,2
AB	3	0,2	GH	4	0,2
BD	4	0,2	FH	5	0,2
BC	5	0,2	EG	6	0,2
AC	6	0,2	EH	6	0,2

Tabelle 6: Sortierte Relationslisten

4.3. Berechnung der Äquivalenzmengen

Gleich erscheinende Relationen werden in Äquivalenzmengen zusammengefasst. Diese sind keine Äquivalenzmengen im mathematischen Sinn, da aufgrund der Toleranzen die Transitivität nicht gegeben ist. Einer Lokalrelation wird dabei eine Menge von Referenzrelationen zugewiesen, wobei zwischen der Lokalrelation und jeder einzelnen dieser Referenzrelationen folgende Bedingungen erfüllt sein müssen:

- Die Typen müssen gleich sein. Eine Punkt–Gerade und eine Punkt–Ebene Relation sind z.B. nicht gleich und somit nicht in derselben Äquivalenzmenge.
- Die Untertypen der Targets der Relationen müssen jeweils gleich sein. Eine Kreis–Kreis und eine Kreis–Kugel Relation können z.B. nicht gleich sein, obwohl es sich bei beiden um eine Punkt–Punkt Relation handelt.
- Eventuell vorhandene zusätzliche Eigenschaften der Targets beider Relationen, wie z.B. die Dicke von Wänden oder der Radius von Kugeln und Zylindern müssen (innerhalb der Fehlertoleranz) gleich sein.
- Die Skalare der beiden Relationen müssen ähnlich sein. Genauer: Die Differenz der Skalare muss vom Betrag her kleiner oder gleich sein als die Summe ihrer Toleranzen: $\left|s_{ref} - s_{lok}\right| \leq \left(\delta_{ref} + \delta_{lok}\right)$. Wenn z.B. $t_{err} = 0{,}1$ ist, dann ist die Relation aus zwei Punkten mit Abstand 3,5 und eine mit Abstand 3,8 als ähnlich anzusehen, da $|3{,}8 - 3{,}5| \leq (2 \cdot 0{,}1 + 2 \cdot 0{,}1)$, also $0{,}3 \leq 0{,}4$.
- Alle Parallelabstände müssen innerhalb der Toleranz ähnlich sein.

Beispiel

Lokal-relation	**Skalar**	**Referenz-relation**	**Skalar**
AD	2	FG	2
CD	2	FG	2
BD	4	EF	4
		GH	4
BC	5	FH	5
AC	6	EG	6
		EH	6

Tabelle 7: Äquivalenzmengen

Die Skalare von BC und FH unterscheiden sich z.B. weniger als die Summe ihrer Toleranzen, deshalb bilden sie gemeinsam eine Äquivalenzmenge.

4.4. Initialisierung der Ähnlichkeitsmatrix

Aufbau der leeren Matrix

Aus allen Targets, die in den Äquivalenzmengen vorkommen, wird eine einzige Matrix mit leeren Einträgen aufgebaut. Dadurch werden bereits alle Targets ausgeschlossen, zu denen keine korrespondierende Relation gefunden werden konnte. Jedes Lokaltarget bezeichnet eine Zeile, jedes Referenztarget eine Spalte der Matrix. Jeder Matrixeintrag besitzt zwei Felder:

- Eine Zahl sim_{rc}, die die Ähnlichkeit zwischen den jeweiligen Targets ausdrückt.
- Eine Liste von Verweisen auf andere Einträge der Matrix, die später für ihre Neuberechnung benötigt werden.

Beispiel

Alle in den Äquivalenzmengen vorkommenden Lokaltargets sind A, B, C und D, diese bezeichnen die Zeilen der Matrix. Alle in den Äquivalenzmengen vorkommenden Referenztargets sind E, F, G, und H, welche die Spalten darstellen.

Setzen initialer Ähnlichkeitswerte

Am Anfang sind alle Lokaltargets zu allen Referenztargets als gleich ähnlich zueinander anzusehen, deswegen werden alle Ähnlichkeitswerte der Matrix auf 1 gesetzt. Bei Matrixeinträgen, zu denen die beiden Targets verschiedenen Typen angehören, wird der Ähnlichkeitswert jedoch auf 0 gesetzt, da diese sicher keine Korrespondenz bilden. So korrespondiert z.B. ein Punkttarget in der Lokalwolke sicher nicht mit einem Ebenentarget in der Referenzwolke.

Beispiel

Target / Target	**E**	**F**	**G**	**H**
A	1	1	1	1
B	1	1	1	1
C	1	1	1	1
D	1	1	1	1

Tabelle 8: Initialisierung der Ähnlichkeitswerte

Alle Kombinationen zwischen den Lokal- und Referenztargets erhalten in diesem Beispiel einen initialen Ähnlichkeitswert von 1, da es sich bei Punkt–Punkt Relationen prinzipiell um mögliche Korrespondenzen handelt.

Setzen der Verbindungen

Jedem Matrixeintrag wird eine Liste von Verbindungen zu anderen Einträgen der Matrix zugewiesen. Diese Listen werden später für die iterierte Neuberechnung der Ähnlichkeitswerte benötigt. Bei Matrixeinträgen, deren Targettypen unterschiedlich sind, ist diese Liste jeweils leer. Andernfalls wird die Liste so aufgebaut, dass die Definition der Ähnlichkeit eines Lokaltargets zu einem Referenztarget angewandt wird. Zwei Targets sind zueinander ähnlich, wenn ihre Nachbartargets als zueinander ähnlich gelten. Hierfür werden jedoch nicht alle oder irgendwelche Nachbarn genommen, sondern nur diejenigen, die gemeinsam in einer Äquivalenzmenge sind.

Beispiel

Wie in Tabelle 7 ersichtlich ist, befinden sich AD und FG in einer Äquivalenzmenge, d.h. diese zwei Relationen sind sehr ähnlich. Der Ähnlichkeitswert zwischen A und F soll sich später neu berechnen aus dem Ähnlichkeitswert zwischen D und G, welches die zwei Nachbarn von A und F sind, zu denen ähnliche Relationen bestehen.
Ein weiteres Beispiel ist die Berechnung der Liste von CG. Wie in Tabelle 7 ersichtlich ist, gibt es zwei Äquivalenzmengen, in denen C auf der Lokalseite und G auf der Referenzseite derselben Menge vorkommt, nämlich CD–{FG} und AC–{EG, EH}. Als äquivalente Nachbarn von CG ergeben sich somit aus der ersten Menge DF und aus der zweiten AE. Um also später den neuen Ähnlichkeitswert zwischen C und G zu bestimmen, wird der alte Ähnlichkeitswert zwischen D und F zu dem zwischen A und E addiert. Listeneinträge, bei denen gleiche Targets mehrmals vorkommen, werden arithmetisch gemittelt.

Target Target	E	F	G	H
A	½CG+½CH	DG	DF+CE	CE
B	DF	DE+CH	DH	DG+CF
C	½AH+½AG	DG+BH	DF+AE	BF+AE
D	BF	½AG+½CG+BE	½AG+½CG+BH	BG

Tabelle 9: Verbindungen zwischen Matrixeinträgen

4.5. Iterierte Neuberechnung der Ähnlichkeitsmatrix

In diesem Schritt werden alle Ähnlichkeitswerte der Matrix neu berechnet. Der neue Ähnlichkeitswert eines Matrixeintrages ist die Summe aller alten Ähnlichkeitswerte von den in der jeweiligen Liste stehenden Einträgen. Für jeden Matrixeintrag wird also für jedes Listenelement dieses Matrixeintrages der alte Ähnlichkeitswert nachgeschaut und für den neuen Wert berücksichtigt.

Diese komplette Neuberechnung der Matrix findet mehrmals statt. Wie oft genau, wird durch einen Parameter festgelegt, der die Qualität des Gesamtergebnisses beeinflusst. Eine ausführliche Betrachtung dieser und anderer Einflussgrößen findet in Kapitel 6: Analysen statt. Ein erster Richtwert ist hier drei, das bedeutet die Matrix wird dreimal neu berechnet.

Beispiel

Der neue Ähnlichkeitswert zwischen C und G wird z.B. berechnen, indem für jede Verbindung dieses Matrixeintrages, also DF und AE, der Wert in der alten Tabelle nachgeschaut wird. Beide Werte sind hier 1, die Summe ist also der neue Ähnlichkeitswert von CG 1+1=2. So wird mit jedem Matrixeintrag verfahren.
Genauso geht es in den nächsten Iterationsschritten weiter: Um z.B. im II. Schritt den Wert zwischen C und H zu berechnen, wird für jeden Verbindungseintrag von CH, also BF und AE, der Wert aus dem I. Schritt nachgeschaut. Bei BF ist dieser 2, bei AE 1. Die Summe ist also 2+1=3.

I.					**II.**					**III.**				
	E	**F**	**G**	**H**		**E**	**F**	**G**	**H**		**E**	**F**	**G**	**H**
A	1	1	2	1	**A**	2	2	3	1	**A**	3	4	4,5	1,5
B	1	2	1	2	**B**	2	3	1	4	**B**	3	5	1	8
C	1	2	2	2	**C**	1,5	4	3	3	**C**	2	8	5	5
D	1	2	2	1	**D**	2	3	4	1	**D**	3	5	7	1

Tabelle 10: Dreifache Neuberechnung der Ähnlichkeitsmatrix

4.6. Bestimmung der Korrespondenzen

Bestimmung der Korrespondenzkandidaten

Ein Eintrag der Ähnlichkeitsmatrix in der Zeile r (row) und der Spalte c (column) gilt als möglicher Kandidat für eine Korrespondenz, sofern er einen relativ hohen Ähnlichkeitswert sim_{rc} hat. Ab wann das der Fall ist, wird mit den zwei Parametern TOTAL_MAX_THRESH und LINE_MAX_THRESH festgelegt. Zunächst werden hierfür alle Zeilenmaxima $rowMax_r$, alle Spaltenmaxima $colMax_c$ und das totale Maximum der Matrix $totalMax$ bestimmt. Ein Eintrag ist genau dann ein Korrespondenzkandidat, wenn folgende drei Bedingungen erfüllt sind:

- $sim_{rc} > \text{TOTAL_MAX_THRESH} \cdot totalMax$
- $sim_{rc} > \text{LINE_MAX_THRESH} \cdot rowMax_r$
- $sim_{rc} > \text{LINE_MAX_THRESH} \cdot colMax_c$

Das bedeutet, dass der Ähnlichkeitswert groß genug sein muss im Vergleich zum totalen Maximum, zum Maximum aller Werte derselben Zeile und zum Maximum aller Werte derselben Spalte.

Beispiel

Seien hier die Parameter LINE_MAX_THRESH = 0,7 und TOTAL_MAX_THRESH = 0,5.

Zeilenkandidaten					**Spaltenkandidaten**					**Kandidaten**				
	E	**F**	**G**	**H**		**E**	**F**	**G**	**H**		**E**	**F**	**G**	**H**
A	3	**3,5**	**4,5**	1,5	**A**	**3**	3,5	4,5	1,5	**A**	3	3,5	4,5	1,5
B	3	5	1	**7,5**	**B**	**3**	5	1	**7,5**	**B**	3	5	1	**7,5**
C	2	**7,5**	5	5	**C**	2	**7,5**	**5**	5	**C**	2	**7,5**	5	5
D	3	**5**	**7**	1	**D**	**3**	5	**7**	1	**D**	3	5	**7**	1

Tabelle 11: Kandidatenfindung durch Maximumvergleich

Alle Einträge der Zeile A, die einen größeren Wert als 0,7 · 4,5 = 3,15 haben, sind Zeilenkandidaten. Für Zeile B liegt der Wert analog bei 0,7 · 7,5 = 5,25. Bei Spalte E ist ein Eintrag Spaltenkandidat, sobald er größer ist als 0,7 · 3 = 2,1. Bei Spalte G sind es 0,7 · 7 = 4,9. Die zusätzliche Bedingung, dass der Eintrag auch noch groß genug relativ zum totalen Maximum der Tabelle sein muss, ist hier im Beispiel für keinen Eintrag ein Problem, denn es gilt 7,5 > 0,5 · 7,5 und 7 > 0,5 · 7,5. Die Kandidaten sind somit BH, CF und DG.

Wahl der besten Kandidaten

Im letzten Schritt wird für alle sinnvollen Untermengen der Kandidatenmenge eine Testregistrierung durchgeführt. Somit fließen jetzt wieder die Positions- und Richtungsangaben der Targets direkt in die Berechnung mit ein. Die Kombination, die möglichst viele Korrespondenzen mit möglichst gutem Ergebnis der Testregistrierung vereint, wird als finale Lösung der Korrespondenzsuche ausgegeben.

Bei jeder möglichen Untermenge der ermittelten Kandidatenmenge wird mit folgenden Kriterien überprüft, ob diese Untermenge eine neue beste Lösung ist:

- Die Menge muss mindestens drei Kandidaten enthalten. Eine Registrierung mit weniger als drei Targets ist ohnehin nicht möglich.
- In der Menge darf kein Target doppelt vorkommen, da es bei zwei Scans keinen Sinn macht, einem Target mehrere korrespondierende Targets zuzuweisen.
- Die Anzahl der Kandidaten der Menge muss mindestens so groß sein, wie die der bisher besten gefundenen Lösung, sofern eine solche existiert.
- Die Testregistrierung liefert Werte für die Standardabweichungen der Positions- und Richtungsdifferenzen, welche gewisse Grenzen nicht überschreiten dürfen.
- Ist die Anzahl der Kandidaten der Menge gleich groß wie die der bisher besten gefundenen Lösung, so muss das Ergebnis der Testregistrierung besser sein.

Beispiel

Im Beispiel ist die Untermenge, die alle oben genannten Kriterien erfüllt, die Menge selbst. Die finale Lösung ist somit B=H, C=F, D=G. Targets der Lokalwolke können in ihre korrespondierenden Targetnamen der Referenzwolke umbenannt werden. Somit kann eine abschließende Registrierung stattfinden.

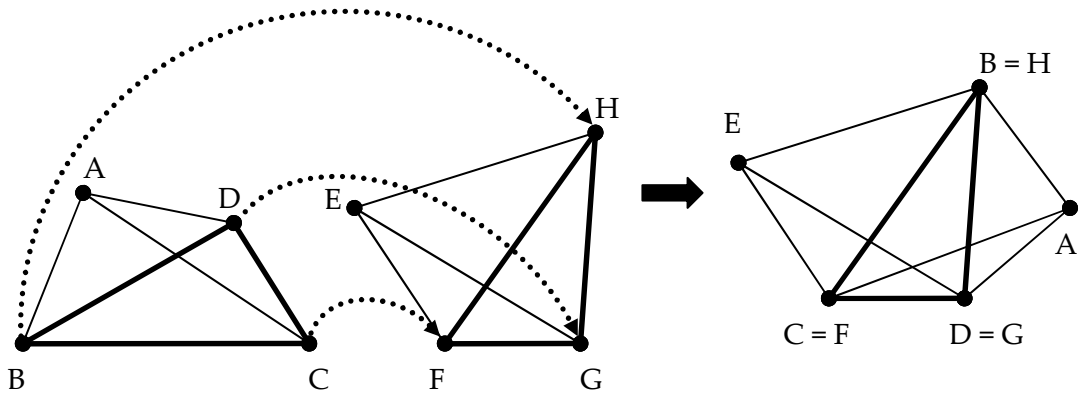


Abbildung 13: Registrierung

5. Varianten

5.1. Gewichtung nach Abweichung

Im dritten Schritt des Algorithmus, bei der Initialisierung der Ähnlichkeitsmatrix, könnten beim Setzen der Verbindungen für einen Eintrag zusätzliche Faktoren als Gewichte berechnet werden. Die Relation, die in die Liste eines Eintrages eingefügt wird, wird dabei mit der zum Eintrag gehörenden Relation verglichen. Je geringer die Differenz zwischen ihren beiden Skalaren ist, desto höher wird das Gewicht gewählt, das dann einen zusätzlichen Faktor für den neuen Listeneintrag darstellt.

Bei dieser Variante gibt es also nicht nur die binäre Entscheidung, ob ein Abstand oder ein Winkel innerhalb der Fehlertoleranz als gleich zu einem anderen Abstand oder Winkel gilt. Stattdessen werden bei der Neuberechnung der Ähnlichkeitsmatrix die Listeneinträge bei kleineren Abweichungen der Skalare stärker und bei größeren Abweichungen weniger stark berücksichtigt.

Beispiel

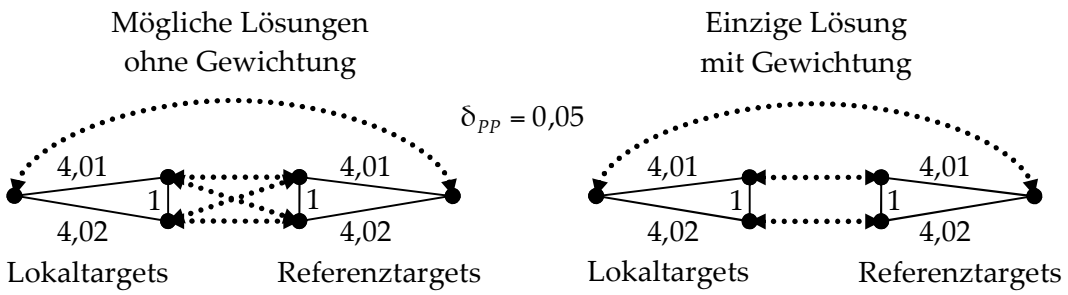


Abbildung 14: Beispiel zur Variante mit Gewichtungen

Das Beispiel zeigt, dass ohne Gewichtung alle Relationen gleichberechtigt sind, die innerhalb der Fehlertoleranz gleiche Skalare haben. Dagegen ist bei der Variante mit Gewichtung die rechnerisch beste Lösung gleichzeitig die einzige.

Fazit

Mathematisch gesehen ist die Variante sinnvoll, da die Lösung bevorzugt wird, bei der die Differenzen der Skalare korrespondierender Relationen geringer sind. Der Mehraufwand zur Berechnung und Anwendung der Faktoren lohnt sich in der Praxis jedoch nicht. Die Unterschiede sind zu gering, schließlich geht es um Größenordnungen der Fehlertoleranz. Außerdem wird im letzten Schritt des Algorithmus sowieso nur die Lösung ausgewählt, die die geringste Abweichung bei der Testregistrierung gezeigt hat, was ein viel genaueres Kriterium ist als hier.

5.2. Eliminierung von Einträgen

Die Variante zur Eliminierung von Einträgen bietet sich an, um die iterierte Neuberechnung der Ähnlichkeitsmatrix zu beschleunigen. Hierbei wird bereits nach wenigen Iterationsschritten in der Matrix nach sehr klaren Korrespondenzen gesucht. Alle anderen Spalten- und Zeileneinträge einer solchen als korrekt eingestuften Korrespondenz werden eliminiert, d.h. ihr Ähnlichkeitswert wird auf 0 gesetzt und ihre Liste zur iterierten Neuberechnung wird gelöscht.

Beispiel

I. Iteration					**II. Iteration**					**Eliminierung**				
	E	**F**	**G**	**H**		**E**	**F**	**G**	**H**		**E**	**F**	**G**	**H**
A	2	2	1	3	**A**	4	3	2	5	**A**	4	**0**	2	5
B	3	2	2	3	**B**	4	4	3	6	**B**	4	**0**	3	6
C	1	5	1	2	**C**	1	**12**	3	4	**C**	**0**	12	**0**	**0**
D	2	1	3	1	**D**	3	3	4	1	**D**	3	**0**	4	1

Tabelle 12: Beispiel zur Eliminierungs-Variante

Im Beispiel wurde nach zwei Schritten wegen der sehr klaren Korrespondenz C=F eliminiert. Nach darauf folgenden drei Iterationen ist das Ergebnis z.B. besser, als wenn man insgesamt fünf Iterationen ohne Eliminierung durchgeführt hätte.

Fazit

Wegen den wenigen Iterationen lohnt sich der Aufwand für die Eliminierung oft nicht. Zumindest bei nicht übermäßig großen Matrizen erhält man ähnlich gute Ergebnisse, wenn man stattdessen am Ende einen Iterationsschritt mehr ausführt.

5.3. Normierung der Matrix

Bei großen Matrizen und vielen Iterationsschritten können die Ähnlichkeitswerte auf hohe Beträge ansteigen, besonders, wenn es viele ähnliche Skalare gibt, z.B. bei sehr vielen Punkttargets mit gleichem Abstand zueinander. Um das zu verhindern, könnte man nach jedem Iterationsschritt eine Normierung der Matrix vornehmen:

$a_{ij,n+1} = \frac{a_{ij,n}}{\|A\|_F}$ mit der in [Matrix96] beschriebenen Frobeniusnorm $\|A\|_F = \sqrt{\sum_{i,j} a_{ij}^2}$

Fazit

Der Aufwand zur Normierung nach jedem Iterationsschritt lohnt sich bei echten Scandaten selten, normalerweise werden die Ähnlichkeitswerte nicht allzu groß.

5.4. Keine Testregistrierung

Es stellt sich die Frage, ob die Testregistrierungen am Ende des Algorithmus nötig sind und ob stattdessen nicht einfach die höchsten Werte der Ähnlichkeitsmatrix die besten Korrespondenzen darstellen. In der Tat ist diese Variante möglich, ohne dass die Ergebnisse der zufallsbasierten Tests wesentlich schlechter werden. Jedoch wird erst bei einer Testregistrierung festgestellt, ob die gefundenen Korrespondenzen auch wirklich gemeinsam realisierbar sind, so gut jede einzelne dabei auch sein mag.

Fazit

Da die Testregistrierung mit geringem Aufwand möglich ist und in einigen Fällen das Ergebnis deutlich besser ausfällt, lohnt es sich nicht, auf sie zu verzichten.

5.5. Ausschließliche Testregistrierung

Umgekehrt könnte eine Korrespondenzsuche auch nur aus Testregistrierungen bestehen. Im Extremfall müssten alle Targetkombinationen durchprobiert werden, wobei diejenige mit der besten Kombination aus vielen Korrespondenzen und gutem Registrierungsergebnis ausgegeben wird. Der Algorithmus wäre dadurch sehr viel einfacher, der Aufwand wäre mit dieser Variante bei sehr kleinen Matrizen geringer. Sobald die Targetmengen etwas größer werden, steigt die Anzahl der durch die Testregistrierung zu überprüfenden Kombinationen jedoch unverhältnismäßig an.

Fazit

Diese Variante wäre sehr einfach zu implementieren und würde auch gute Ergebnisse liefern. Für sehr kleine Targetmengen ist sie sogar die bessere Lösung, bei etwas größeren Matrizen ist der Rechenaufwand jedoch inakzeptabel hoch.

5.6. Inkrementelle Neufindung

Löscht man nach einer Korrespondenzfindung ein Target oder fügt eines hinzu, dann werden bei einer erneuten Korrespondenzfindung alle Daten aus den Targets komplett neu berechnet. Bei dieser Variante hier werden stattdessen so viele Daten wie möglich beibehalten, damit bei einer erneuten Korrespondenzfindung nur die Veränderungen der Targetmengen neu mit eingebracht werden müssen.

Fazit

Ein Vorteil ist eine etwas geringere Rechenzeit bei Änderungen in sehr großen Targetmengen. Nachteile sind ein viel höherer Verwaltungsaufwand für die Daten und damit auch eine kompliziertere Implementierung.

6. Analysen

6.1. Parameter für Targetwolken

Der beschriebene Algorithmus hat viele Einflussgrößen, die das Ergebnis der Korrespondenzfindung maßgeblich beeinflussen. Um die optimale Konfiguration zu finden, werden hier Wolken von Targets zufallsbasiert nach bestimmten Regeln erzeugt und dem Algorithmus als Eingabe übergeben. Der Aufbau dieser Targetwolken kann mit den hier vorgestellten Parametern beeinflusst werden. Für die spätere Anwendung in einer Applikation sind diese Parameter ohne Bedeutung, sie werden nur für die folgenden heuristischen Untersuchungen benötigt. Um deutlichere Ergebnisse zu erhalten, wird bei allen hier durchgeführten Tests auf die Testregistrierung verzichtet.

Parameter	**Erklärung**
POSITION_MAX = 100m	Die Positionsvektoren aller zufällig erzeugten Targets liegen in einem endlichen Bereich, wobei der Betrag jeder Komponente des Vektors nicht größer ist als dieser Parameter. Die Größenordnung der Reichweite eines 3D-Laserscanners ist derzeit etwa 100 Meter, deshalb wird dieser Wert hier verwendet.
JITTER_MAX = 2cm	Bei der zufälligen Erzeugung der Vektoren werden die Werte durch Zufall in einem hier definierten Rahmen verfälscht. Die Messgenauigkeit eines Laserscanners liegt zwar im Millimeterbereich, jedoch könnte eine ungenaue Positionserkennung von Targets schlechtere Werte liefern. Deshalb wird hier zur sicheren Seite hin ein Wert von 2cm angenommen.

Tabelle 13: Parameter zufallserzeugter Targetwolken

Alle untersuchten Parameterkombinationen werden zufallsbasiert mit zwei verschiedenen Testarten ausgeführt:

- Standardtest: Innerhalb der angegebenen Grenzen und Verhältnisse sind hier Position, Lage, Art und Anzahl der Targets dem Zufall überlassen.
- Paralleltest: Um den sehr häufigen Anwendungsfall von Scans in geschlossenen Räumen besser abzudecken, sind hier die Targets absichtlich oft parallel oder orthogonal zueinander ausgerichtet. Als zusätzliche Schwierigkeit ist hier die Anzahl der Targets pro Wolke viel geringer als beim Standardtest.

Parameter	Erklärung
RANDOM_TESTS = 10000	Legt die Anzahl der auszuführenden Standardtests für diese Untersuchung fest.
TARGETS_CLOUD_MIN = 8 TARGETS_CLOUD_MAX = 12	Die Anzahl der Targets für die zufallserzeugte Lokalwolke wird hier durch eine untere und obere Schranke festgelegt.
TARGETS_REM_MIN = 3 TARGETS_REM_MAX = 5	Um die Referenzwolke zu erzeugen, wird eine Kopie der Lokalwolke erstellt, von der durch Zufall Targets entfernt werden. Wie viele das minimal und maximal sind, wird hier festgelegt.
TARGETS_ADD_MIN = 3 TARGETS_ADD_MAX = 5	Zusätzlich werden zur Referenzwolke weitere Targets durch Zufall hinzugefügt. Wie viele das minimal und maximal sind, wird hier festgelegt.
WEIGHT_POINTS = 2 WEIGHT_PIPES = 1 WEIGHT_PLANES = 1	Hier kann durch Gewichtung festgelegt werden, wie das Verhältnis der Anzahl von Punkt-, Zylinder- und Ebenen-Targets im Durchschnitt sein soll.

Tabelle 14: Parameter für den Standardtest

Parameter	Erklärung
PARALLEL_TESTS = 100000	Legt die Anzahl der auszuführenden Paralleltests für diese Untersuchung fest.
P_TARGETS_CLOUD_MIN = 3 P_TARGETS_CLOUD_MAX = 4	Analog zum Standardtest. Hier sollen jedoch viel weniger Targets zur Verfügung stehen.
P_TARGETS_REM_MIN = 0 P_TARGETS_REM_MAX = 1	Analog zum Standardtest. Hier wird aber nur ein oder kein Target entfernt, da es nur wenige gibt.
P_TARGETS_ADD_MIN = 0 P_TARGETS_ADD_MAX = 1	Analog zum Standardtest. Hier wird nur ein oder kein Target hinzugefügt.
P_WEIGHT_POINTS = 1 P_WEIGHT_PIPES = 1 P_WEIGHT_PLANES = 3	Analog zum Standardtest. Hier soll es aber deutlich mehr Ebenen als Punkte und Zylinder geben, damit die Korrespondenzfindung schwieriger wird.
P_PARALLEL = 75%	Wahrscheinlichkeit, mit der ein erzeugtes Target parallel zu dem zuvor erzeugten Target ist. Im Paralleltest soll es viele parallele Targets geben.
P_ORTHOGONAL = 20%	Wahrscheinlichkeit, mit der ein erzeugtes Target rechtwinklig zu dem zuvor erzeugten Target ist. Mit einem Wert von 20% (und 75% bei der Parallelität) bleiben nur 5% für zufällige Richtungen übrig.

Tabelle 15: Parameter für den Paralleltest

6.2. Metriken

Die Parameter für die Targetwolken dienen als Eingabe bei der Untersuchung der optimalen Konfiguration der Parameter für den Algorithmus. Die Bewertung, ob eine bestimmte Ausgabe besser oder schlechter als eine andere ist, kann mit folgenden zwei Metriken stattfinden:

$$k = \frac{\text{Anzahl gefundener Korrespondenzen, die korrekt sind}}{\text{Anzahl aller korrekten Korrespondenzen}},$$

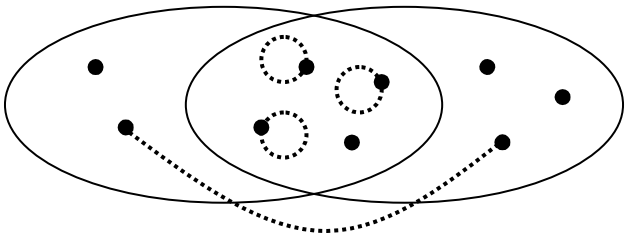
$$i = \frac{\text{Anzahl gefundener Korrespondenzen, die inkorrekt sind}}{\sqrt{\text{Anzahl aller inkorrekten Korrespondenzen}}}.$$

Beide Werte liegen im Intervall $[0..1]$. Idealerweise sollen alle korrekten Korrespondenzen gefunden werden ($k = 1$) und möglichst keine, die inkorrekt sind ($i = 0$). Die Anzahl aller inkorrekten Korrespondenzen setzt sich zusammen aus dem Produkt zwischen der Anzahl Lokaltargets und der Anzahl Referenztargets, abzüglich der korrekten Korrespondenzen. Dieser Wert ist jedoch quadratisch in der Anzahl der Targets, was mit der Wurzel ausgeglichen wird. Zusätzlich ist in allen folgenden Diagrammen der (sehr niedrige) Wert von i um das 20-fache überhöht.

Die Metriken hängen vom kompletten Satz $\vec{P}$ aller beschriebenen Parameter ab und können sowohl für den Standardtest, als auch für den Paralleltest bestimmt werden. Pro Parameterkonfiguration ergeben sich also folgende vier Werte: $k_s(\vec{P}),\ i_s(\vec{P}),\ k_p(\vec{P}),\ i_p(\vec{P})$.

Beispiel:

Eine Lokalwolke mit 6 Targets und eine Referenzwolke mit 7 Targets überlappen sich in insgesamt 4 Targets. Es werden 4 Korrespondenzen gefunden, davon sind jedoch nur 3 korrekt:



$$k = \frac{3}{4} = 0.75$$

$$i = \frac{1}{\sqrt{6 \cdot 7 - 4}} \approx 0.16$$

Abbildung 15: Beispielberechnung der Metriken

6.3. Parameter zur Ähnlichkeitsmatrix

Mit den Parametern für Targetwolken werden Eingabedaten erzeugt, deren Ausgaben durch die beschriebenen Metriken ausgewertet werden. Durch Variation der drei hier beschriebenen Parameter kann das Endergebnis verbessert werden.

Parameter	Erklärung
RECALC_NUM = 5	Dieser Parameter gibt die Gesamtzahl der iterierten Neuberechnungen der Ähnlichkeitsmatrix an.
LINE_MAX_THRESH = 0.6	Der Ähnlichkeitswert muss groß genug sein im Verhältnis zum Maximum der Zeile und der Spalte, in dem sich der Wert befindet.
TOTAL_MAX_THRESH = 0.6	Der Ähnlichkeitswert muss groß genug sein im Verhältnis zum Maximum der Matrix.

Tabelle 16: Parameter zur Ähnlichkeitsmatrix

Zur Veranschaulichung von LINE_MAX_THRESH und TOTAL_MAX_THRESH sind in Abbildung 16 beispielhaft die Ähnlichkeitswerte einer Zeile und das totale Matrixmaximum einer Ähnlichkeitsmatrix dargestellt. Hierfür wurde LINE_MAX_THRESH = 0.6 und TOTAL_MAX_THRESH = 0.5 gewählt.

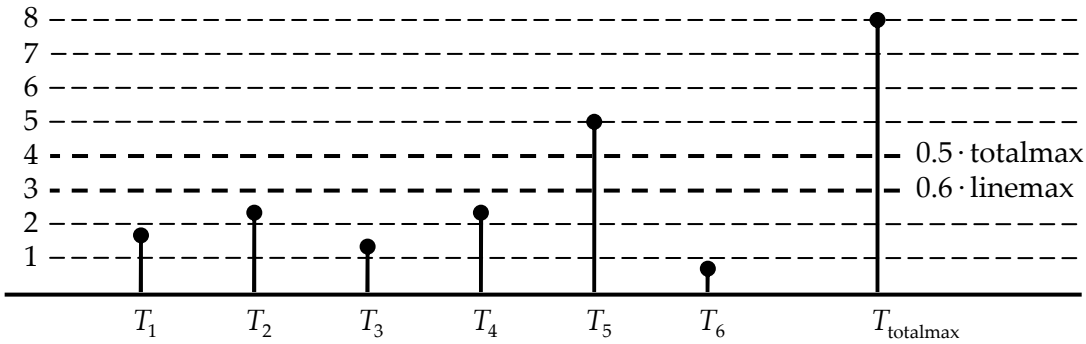


Abbildung 16: Beispiel zu LINE_MAX_THRESH und TOTAL_MAX_THRESH

Untersucht wird hier T_5. Es muss groß genug im Vergleich zum Zeilenmaximum 5 sein. Trivialerweise ist $0.6 \cdot 5 = 3 \leq 5$, da es sich um das Maximum selbst handelt. Zusätzlich darf der Wert im Vergleich zum totalen Maximum nicht zu klein sein, hier also $5 \geq 0.5 \cdot 8 = 4$, was hier auch zutrifft. Die Überprüfung in Bezug auf die Maxima in den Spalten funktioniert völlig analog.

Variation von RECALC_NUM und LINE_MAX_THRESH

RECALC_NUM wird von 1 bis 8 variiert, LINE_MAX_THRESH von 0.0 bis 1.0.

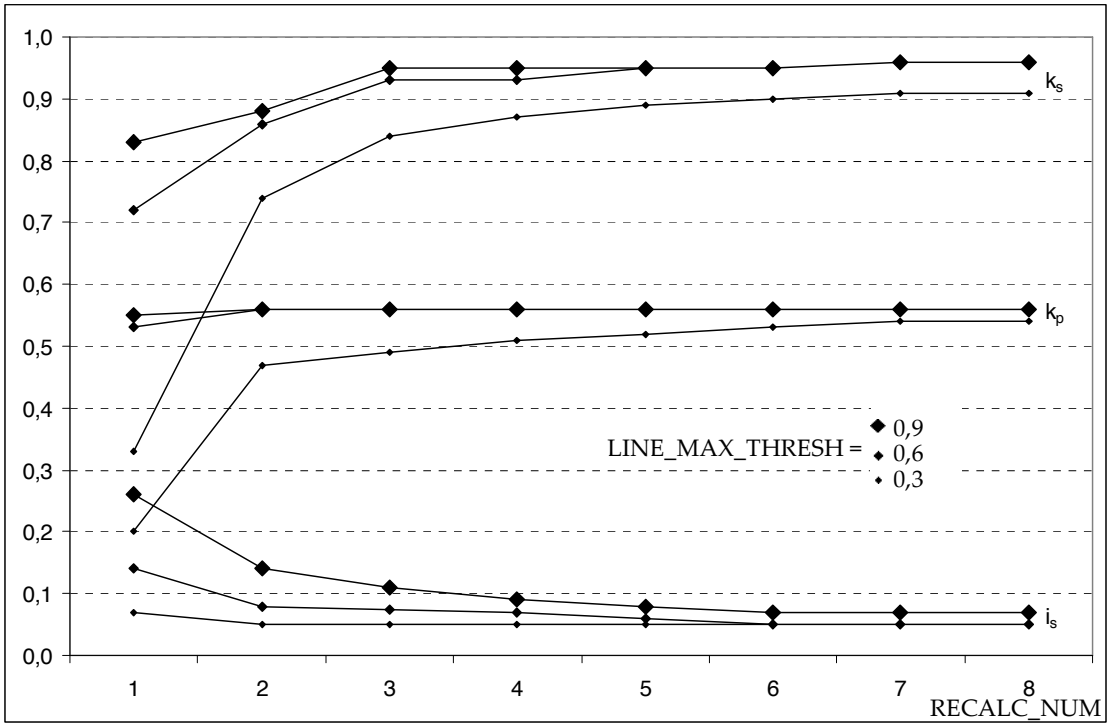


Diagramm 1: Variation von RECALC_NUM und LINE_MAX_THRESH

Ergebnisse:

- Bereits nach einem (!) Iterationsschritt wird im Standardtest über 80% aller korrekten Korrespondenzen gefunden, nach drei Schritten sind es schon 95%.
- Für k gibt es ab dem 3. Schritt keine wesentliche Verbesserung mehr, für i gilt das ab etwa dem 5. Schritt. Ein gute Wahl ist damit RECALC_NUM = 5.
- Der Vorteil eines niedrigeren LINE_MAX_THRESH liegt nur im geringeren Wert von i, der Nachteil ist ein ebenfalls geringeres k.
- Dieser Nachteil ist jedoch ab etwa dem 5. Schritt nicht mehr vorhanden, hier ist im Standardtest k gleich groß für mittlere und große LINE_MAX_THRESH. Für sehr kleine Werte werden jedoch zu viele korrekte Lösungen nicht gefunden, weshalb LINE_MAX_THRESH = 0.6 ein guter Mittelweg ist.
- Die Metrik der inkorrekten Korrespondenzen für den Paralleltest i_p ist nicht angegeben, da sich diese auf konstant sehr niedrigem Niveau befindet.

Variation von TOTAL_MAX_THRESH

TOTAL_MAX_THRESH wird von 0.0 bis 1.0 variiert.

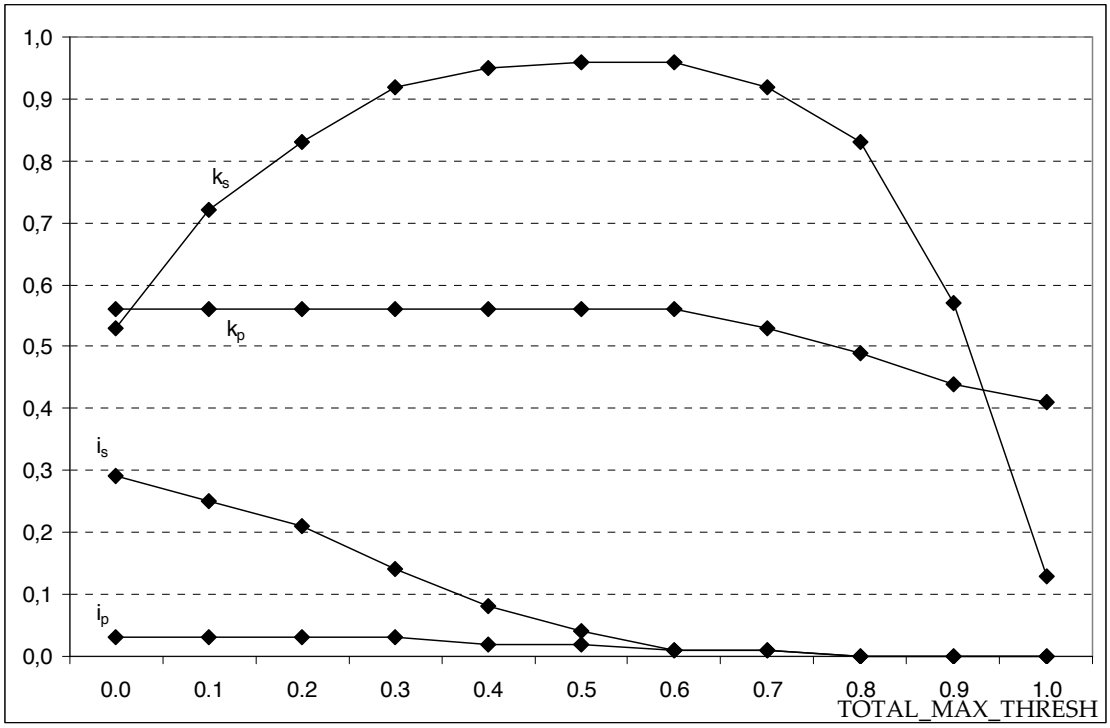


Diagramm 2: Variation von TOTAL_MAX_THRESH

Ergebnisse:

- Auf den ersten Blick ist es überraschend, dass bei einem kleineren Wert von TOTAL_MAX_THRESH weniger Korrespondenzen gefunden werden. Ein kleinerer Wert bedeutet eine weniger strenge Auswahl der Maxima und somit eigentlich mehr korrekte Korrespondenzen. Zunächst ist das auch richtig, jedoch werden dabei auch deutlich mehr falsche Korrespondenzen gefunden, so dass bei der Verifikation am Ende die Testregistrierung oft fehlschlägt. Zur sicheren Seite hin werden dabei keine gefundenen Korrespondenzen ausgegeben, was insgesamt zu einem geringeren Wert von k führt.
- Der ideale Wert für TOTAL_MAX_THRESH ist 0.6. Für kleinere Werte werden eindeutig zu viele falsche Lösungen gefunden, vor allem beim Standardtest. Für größere Werte als 0.6 sinkt der Anteil korrekter Korrespondenzen zu stark ab.
- Die Auswirkungen auf den Paralleltest sind gering, da durch die vielen Parallelitäten und die wenigen Targets die Maxima ähnlich groß sind.

6.4. Parameter zur Toleranz

Diese Parameter definieren Schranken, bei deren Überschreitung der jeweilige Wert nicht mehr berücksichtigt wird. Bleibt der Wert jedoch innerhalb der Schranke, so wird er toleriert, was z.B. bei kleinen Ungenauigkeiten in den gemessenen Werten der Fall ist.

Parameter	Erklärung
TRANSLATION_ERROR_MAX = 2cm	Maximal akzeptierter Translationsfehler im Positionsvektor eines Targets. Aus denselben Gründen wie bei JITTER_MAX wird der Wert hier auf 2cm gesetzt.
ROTATION_ERROR_MAX = 1°	Maximal akzeptierter Rotationsfehler im Lagevektor eines Nicht-Punkt-Targets. Für einen auf Länge 1 normierten Lagevektor mit einem Translationsfehler an der Spitze von 2cm ist der Rotationsfehler etwa 1°.
POSITION_ACCURACY_MAX = 4cm ANGLE_ACCURACY_MAX = 2°	Der bei der Verifikation der Lösung maximal akzeptierte Fehler in den Positionen und den Lagewinkeln. Pauschal werden sie hier auf das doppelte der obigen Parameter gesetzt.
PARALLEL_ANGLE_MAX = 20°	Maximaler Winkel zwischen zwei Nicht-Punkt-Targets, bei dem ihr Abstand zueinander berücksichtigt werden soll (siehe Abschnitt 3.3: Abstände und Winkel – Parallelabstand).
PARALLEL_DIST_DIFF = 2m	Wurde bei zwei Relationen jeweils der Abstand ihrer beiden Nicht-Punkt-Targets berücksichtigt, dann ist der maximal akzeptierte Unterschied in den beiden Abständen hier definiert. Der relativ hohe Wert ergibt sich aus der Mehrdeutigkeit der Positionsvektoren.

Tabelle 17: Parameter zur Toleranz

Die Werte der Parameter TRANSLATION_ERROR_MAX, ROTATION_ERROR_MAX, POSITION_ACCURACY_MAX und ANGLE_ACCURACY_MAX ergeben sich mehr oder weniger direkt aus der Genauigkeit der Scandaten, somit ist ihre Variation weniger interessant. Tendenziell sollten sie eher niedrig gewählt werden, da weniger korrekte Korrespondenzen nicht so schlimm sind wie mehr falsche Korrespondenzen.

Interessanter ist der Einfluss von PARALLEL_ANGLE_MAX und PARALLEL_DIST_DIFF, besonders auf den Paralleltest.

Wahl von PARALLEL_DIST_MAX

Die Wahl dieses Parameters gestaltet sich für den allgemeinen Fall als schwierig. Abhängig davon, wie sehr sich die Positionspunkte identischer Targets von Scan zu Scan unterscheiden, kann es nötig sein, sich für ein sehr großes PARALLEL_DIST_MAX zu entscheiden. Eine bewährte Methode ist es, den Schwerpunkt des Ebenen- bzw. Geradenausschnittes als Positionspunkt zu verwenden. Es kann davon ausgegangen werden, dass in den allermeisten Fällen der Abstand von Schwerpunkten identischer Targets nicht mehr als etwa 10 Meter beträgt. Geht man weiterhin davon aus, dass die Targets eher parallel als rechtwinklig zueinander sind, dann ist die Wahl von PARALLEL_DIST_MAX = 2m für diese Analyse ein vernünftiger Wert.

Variation von PARALLEL_ANGLE_MAX

PARALLEL_ANGLE_MAX wird in 10°-Schritten von 0° bis 90° variiert. Zusätzlich gibt es einen Fall mit dem fiktiven Winkel von -10°, damit in der Analyse auch die Ergebnisse für eine Deaktivierung des Parallelabstandes berücksichtigt werden.

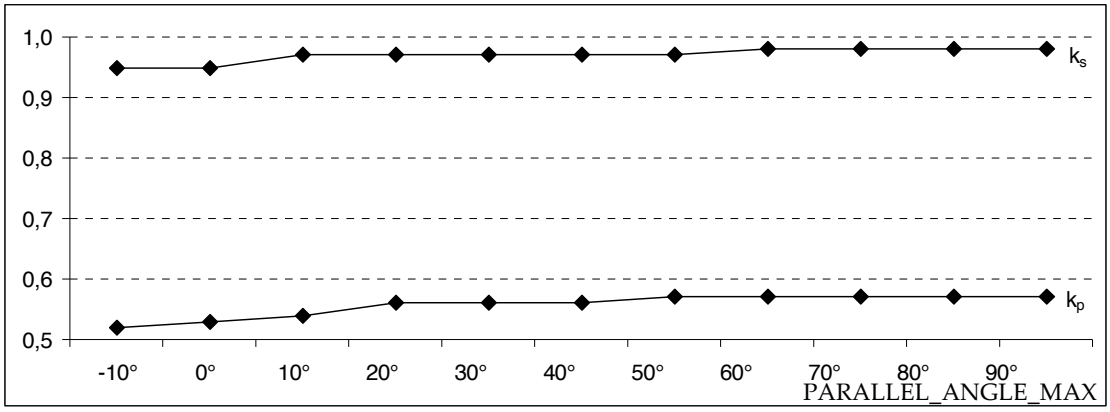


Diagramm 3: Variation von PARALLEL_ANGLE_MAX

Ergebnisse:

- i_s und i_p sind konstant sehr niedrig und deswegen hier uninteressant.
- Wie erwartet ist der Einfluss auf den Standardtest nicht wesentlich, da hier die Targets viel seltener parallel zueinander sind.
- PARALLEL_ANGLE_MAX = 20° ist eine gute Wahl, für größere Werte gibt es kaum eine Verbesserung.
- Es ist schlecht, einen zu großen Winkel zu nehmen, da sich dann eine Variation des Positionspunktes viel deutlicher auf den Parallelabstand auswirkt.

6.5. Robustheit

Es kann sein, dass die Fehler in den Positionen und Winkeln der Targets so groß werden, dass die angegebenen Parameter zur Toleranz nicht mehr ausreichen. Die Frage hierbei ist, wie robust der Algorithmus mit dem Konzept der Ähnlichkeitsmatrix in solchen Fällen reagiert. Um das zu untersuchen, wird ein Teil der Relationen direkt nach ihrer Erzeugung wieder künstlich gelöscht.

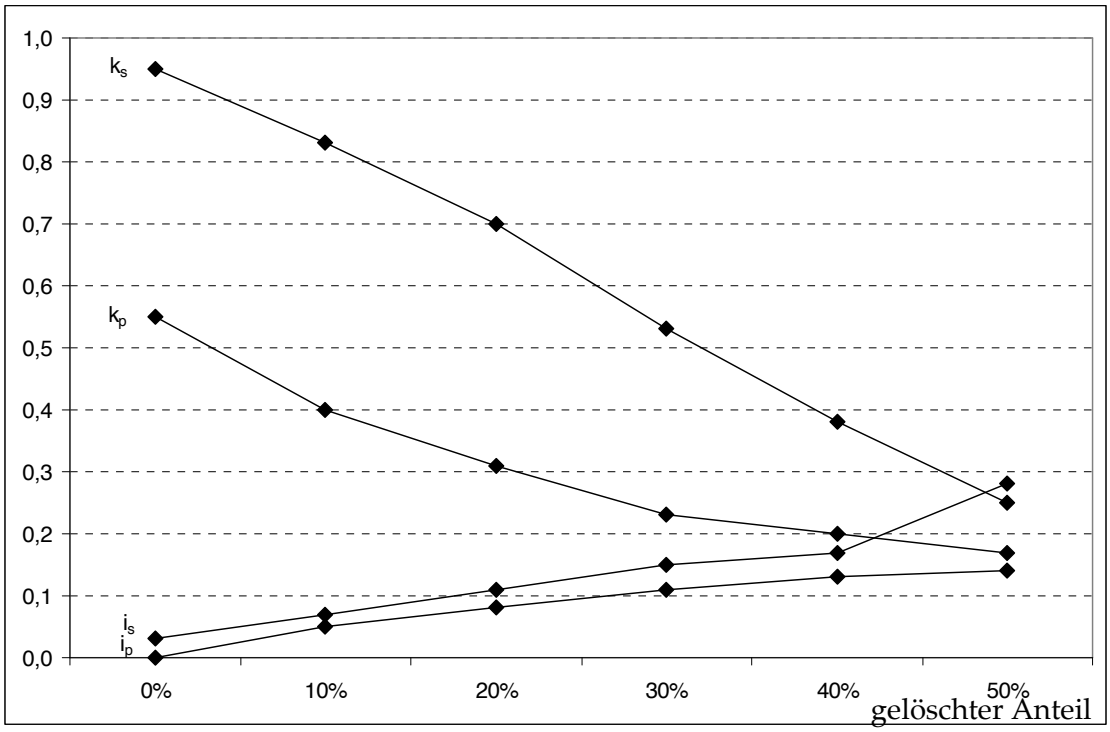


Diagramm 4: Künstliches Löschen von Relationen

Ergebnisse:

- Der Algorithmus ist als robust anzusehen, selbst bei Löschung von 30% der Relationen werden im Standardtest noch über 50% der korrekten Korrespondenzen gefunden.
- Durch die geringere Anzahl an Targets und Relationen im Paralleltest wirkt sich dort eine Löschung etwas stärker aus. Trotzdem werden hier bei einer Löschung von 40% noch 20% der korrekten Korrespondenzen gefunden.
- Der Anteil falscher Korrespondenzen steigt zwar mit dem Anteil der Löschung, er bleibt jedoch trotzdem klein. Selbst bei einer Löschung von jeder zweiten Relation liegt er bei unter 1,5% (wie immer ist die Metrik i 20-fach überhöht).

7. Theorie

7.1. Definitionen

Eine Korrespondenz sei „maximal“, wenn sie die größtmöglichste Anzahl an Targets enthält, sie sei „korrekt“, wenn jede ihrer Targetzuordnungen tatsächlich identischen Targets entspricht und sie sei „konsistent“, wenn theoretisch (basierend auf den Positionen und Richtungen der Targets) jede ihrer Targetzuordnungen identischen Targets entsprechen könnte.

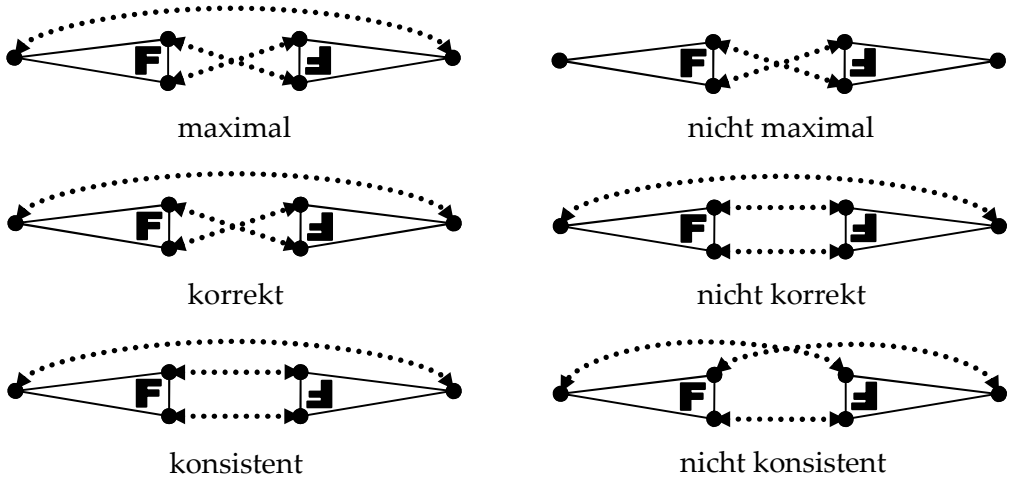


Abbildung 17: Klassifikation von Korrespondenzen

Eine konsistente maximale Korrespondenz zwischen zwei Targetmengen sei genau dann „eindeutig“, wenn es genau eine Menge mit maximaler Anzahl an Targets gibt, die Teilmenge beider Targetmengen ist. Diese Menge enthält alle Targets, die in beiden Scans vorhanden sind, sie steht somit für deren Überlappung. Gibt es mehrere solcher Mengen mit gleich vielen Targets, dann ist die Korrespondenz „mehrdeutig“.

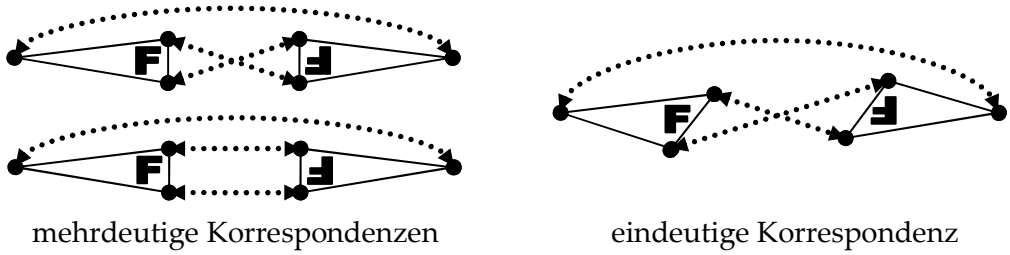


Abbildung 18: Mehr- und Eindeutigkeit von Korrespondenzen

7.2. Korrektheit

Satz

Gibt es zwischen zwei Targetmengen mit l Lokaltargets und r Referenztargets eine maximale konsistente Korrespondenz, die zusätzlich eindeutig ist, dann wird diese von dem in dieser Arbeit vorgestellten Algorithmus spätestens nach k vielen Neuberechnungen der Ähnlichkeitsmatrix gefunden.

Ist eine Korrespondenz maximal und konsistent, jedoch mehrdeutig, dann wird nach spätestens k vielen Neuberechnungen der Ähnlichkeitsmatrix entweder diese Korrespondenz oder eine andere maximale konsistente Korrespondenz gefunden, die gleich viele Targets hat. In beiden Fällen gilt $k = \min(l, r)$.

Beweis

Bei der ersten Neuberechnung der Ähnlichkeitsmatrix wird jedem Paar aus Lokal- und Referenztarget die Anzahl der als gleichwertig geltenden Relationen dieser beiden Targets als Ähnlichkeitswert zugewiesen.

Sieht man die Targetwolken als Graphen, deren Knoten die Targets und deren Kanten die Relationen sind, dann kann der neue Wert, der in der zweiten Iteration einem Paar von Targets (Knoten) zugewiesen wird, folgendermaßen beschreiben werden: Von jedem Knoten des Paares wird jeweils eine Breitensuche in seinem Graphen durchgeführt, hier jedoch zunächst nur bis zur Tiefe zwei (da dies momentan die zweite Iteration ist). Die Anzahl gleichwertiger Pfade der beiden Suchbäume von der Wurzel bis zu den Blättern ist der neue Wert des Knotenpaares. Zwei Pfade gelten dabei als gleichwertig, wenn alle ihre Kanten gleichwertig sind. Zwei Kanten gelten als gleichwertig, wenn die entsprechenden Relationen in einer Äquivalenzmenge sind.

Der dritte Schritt ist analog zum zweiten, allerdings gehen die beiden Suchbäume hier bis zur Tiefe drei. Der neue Wert ist wieder die Anzahl gleichwertiger Pfade. Genauso folgen auch alle weiteren Iterationsschritte, in denen die Tiefe jeweils um eins erhöht wird.

Gesucht ist der maximale gemeinsame Teilgraph, dieser stellt die Überlappung beider Graphen und zugleich die eindeutige (oder eine der mehrdeutigen) maximalen korrekten Korrespondenzen dar. Dieser Teilgraph kann nicht mehr Knoten haben, als die Knotenzahl des kleineren der beiden Graphen, also $\min(l, r)$. Nach genau so vielen Iterationen ist also garantiert, dass alle Kanten dieses Teilgraphen in allen Suchbäumen komplett vorkommen.

Nur Pfade der beiden Graphen, die sich vollständig innerhalb eines (nicht notwendigerweise des größten) Teilgraphen befinden, sind durch den Algorithmus als gleichwertig beurteilt. Der größte dieser Graphen hat die meisten gleichwertigen Kanten und somit auch die meisten gleichwertigen Pfade in beiden Suchbäumen.

Der Algorithmus gibt diejenigen Paare von Targets aus, die den größten Wert in der Matrix haben, die also die größte Anzahl an gleichen Pfaden in ihren Suchbäumen besitzen, was genau dem größten gemeinsamen Teilgraphen entspricht.

Angenommen, es gäbe eine Korrespondenz B, die gleich viele oder mehr Targets als die vom Algorithmus gefundene Korrespondenz A hätte. In diesem Fall gäbe es auch mehr übereinstimmende Pfade in den Suchbäumen von B als in denen von A. Daraus würde folgen, dass die Einträge in der Matrix für Korrespondenz B größer sind als die von A. Damit hätte der Algorithmus aber nicht A als Maximum ausgegeben, woraus folgt, dass diese Korrespondenz eine maximale sein muss.

Beispiel

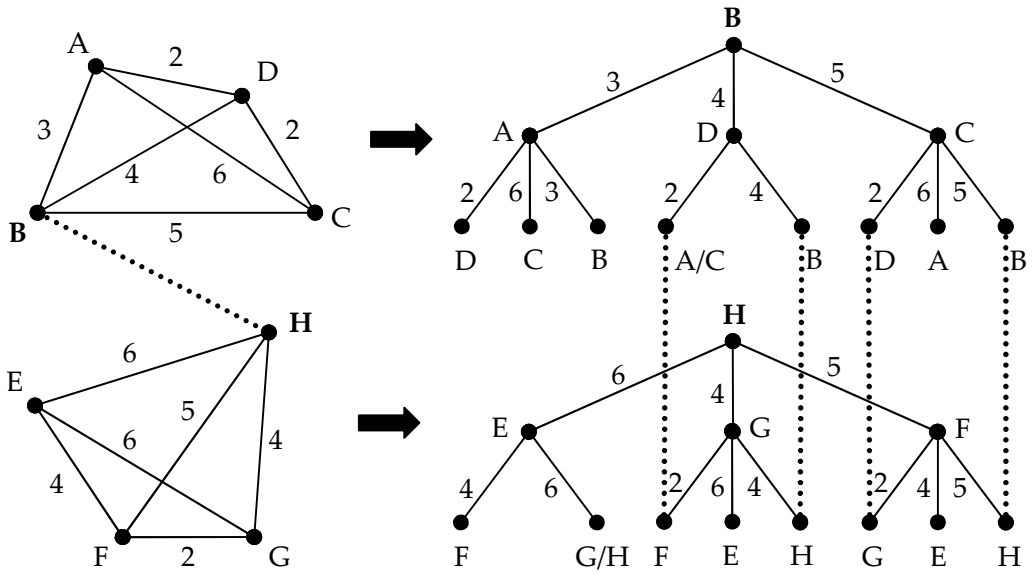


Abbildung 19: Beispiel zum Beweis

Hier im Beispiel sind die aus dem Lokal- und Referenzgraphen entstandenen Suchbäume für den Matrixeintrag B–H nach der zweiten Iteration abgebildet. Alle in beiden Bäumen vorkommenden Pfade sind 4–2, 4–4, 5–2 und 5–5. Die Blätter dieser Pfade sind A/C=F, B=H, D=G und nochmals B=H. Diese stimmen bereits jetzt erstaunlich gut mit den korrekten Korrespondenzen C=F, B=H und D=G überein.

7.3. Komplexität

Voraussetzungen

Folgende Annahmen werden für die Untersuchung der Komplexität gemacht, insbesondere treffen diese bei vernünftigen echten Scandaten zu:

- In der Menge aller Referenzrelationen ist die Anzahl der Relationen mit ähnlichem Skalar linear in der Anzahl der Referenztargets.
- Mit dieser Annahme kann man voraussetzen, dass es pro Lokalrelation höchstens konstant viele Referenzrelationen mit ähnlichem Skalar gibt.
- Die Anzahl der Referenztargets r ist von der gleichen Größenordnung wie die Anzahl der Lokaltargets l.
- Die Anzahl der durchzuführenden Testregistrierungen wird nach oben hin durch eine sehr große konstante Zahl beschränkt (Deckelung). Theoretisch könnte man den gleichen Effekt auch durch eine strengere Wahl der Parameter erhalten, wodurch es auch weniger Korrespondenzkandidaten geben würde.

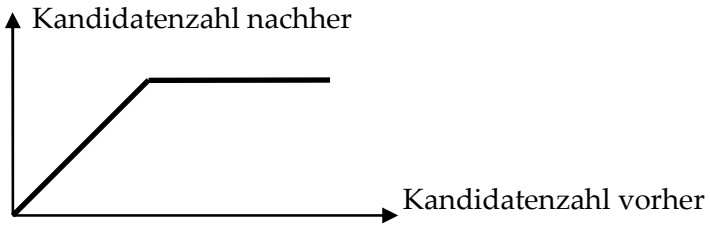


Abbildung 20: Deckelung der Korrespondenzkandidaten

Platzaufwand

Der größte Platz wird für die Ähnlichkeitsmatrix gebraucht. Jeder Eintrag benötigt mit den genannten Annahmen konstant viel Speicher. Es gibt insgesamt $l \cdot r$ viele Einträge, wobei l und r von derselben Größenordnung sind. Damit liegt der Platzaufwand bei $O(r^2)$. Alle anderen Speicherelemente, wie z.B. die Listen der Targets und auch die Listen mit den Äquivalenzrelationen haben einen gleich hohen oder geringeren Bedarf und liegen somit innerhalb dieser Größenordnung.

Die typische Anzahl von Targets bei echten Scans liegt weit unterhalb dessen, was wegen zu viel benötigten Speicherplatzes kritisch werden könnte. Hinzu kommt, dass nach jeder Korrespondenzsuche alle dazu gehörenden Speicherobjekte bedenkenlos gelöscht werden können. Die gefundenen Korrespondenzen bleiben dabei über die gleichen Namen korrespondierender Objekte erhalten. Der Platzbedarf ist also nicht als kritisch einzustufen.

Zeitaufwand

Die Zeitkomplexität für eine Korrespondenzfindung beträgt unter den gemachten realistischen Annahmen im schlechtesten Fall $O(r^2 \cdot \log(r))$, wobei r die Anzahl der Referenztargets ist. Die einzelnen Schritte haben folgenden Zeitaufwand:

1. Berechnung der Relationen – $O(r^2 \cdot \log(r^2) + l^2 \cdot \log(l^2))$

   Für r Referenz- und l Lokaltargets werden jeweils zwei Targets miteinander zu einer Relation verbunden. Es gibt somit r^2+l^2 viele Relationen, die jeweils mit konstantem Aufwand berechnet werden können. Das Sortieren der Listen mit den Längen r^2 und l^2 benötigt dann $r^2 \cdot \log(r)^2$ und $l^2 \cdot \log(l)^2$ viel Zeit.

2. Berechnung der Äquivalenzmengen – $O(r^2+l^2)$

   Da beide Listen jeweils nach ihren Skalaren sortiert sind, können diese parallel durchlaufen werden. Manchmal kann dabei ein Rücksprung zu einem vorigen Eintrag nötig ist, z.B. wenn es zwei Lokaltargets und zwei Referenztargets jeweils mit den Skalaren 4,01 und 4,02 gibt. Dann werden beiden Lokaltargets jeweils beide Referenztargets zugewiesen, wodurch in der Referenzliste ein Rücksprung nötig ist. Das passiert jedoch nicht öfter als es Listeneinträge gibt, wodurch das Durchgehen insgesamt linear in der Anzahl der Relationen bleibt. Bei Listenlängen von r^2+l^2 ist somit der Zeitaufwand genauso groß.

3. Initialisierung der Ähnlichkeitsmatrix – $O(r \cdot l)$

   Die Ähnlichkeitsmatrix hat $r \cdot l$ viele Einträge, für jeden ist ein konstanter Berechnungsaufwand nötig. Insbesondere ist die Anzahl der Verbindungen pro Eintrag aufgrund der gemachten Voraussetzungen konstant.

4. Iterierte Neuberechnung der Ähnlichkeitsmatrix – $O(r \cdot l)$

   Die Neuberechnung der $r \cdot l$ vielen Einträge findet konstant oft statt, pro Eintrag ist wie beschrieben auch nur ein konstanter Aufwand nötig.

5. Bestimmung der Korrespondenzen – $O(r \cdot l)$

   Bei der Bestimmung der Korrespondenzkandidaten muss die komplette Matrix mit ihren $r \cdot l$ vielen Einträgen zweimal durchlaufen werden. Einmal, um das totale Maximum und alle Zeilen- und Spaltenmaxima zu erhalten, und einmal, um festzustellen, welche Einträge diese Grenzwerte einhalten. Wegen der Deckelung der Anzahl der Testregistrierungen wird der Aufwand von konstanten Vielfachen von $r \cdot l$ nicht überschritten.

Da die Anzahl der Targets in Lokal- und Referenzwolke von der gleichen Größenordnung sind, ergibt sich ein Gesamtaufwand von $O(r^2 \cdot \log(r^2))$.

7.4. Matching Algorithmen

Graph Matching

Aus den Targetwolken werden Datenstrukturen aufgebaut, die in Form von Graphen vorliegen. Um ähnliche Muster in den verschiedenen Graphen zu finden, können vorhandene Matching Algorithmen verwendet werden.

In [GrMat96] ist ein Graph Matching Verfahren vorgestellt, mit dem gleiche Strukturen in zwei verschiedenen Graphen gefunden werden können. Der Algorithmus ist robust gegenüber kleineren Abweichungen und ist nicht auf eine spezielle Klasse von Graphen beschränkt.

Aus den Adjazenzmatrizen der beiden Graphen wird eine Matrix, die sogenannte Match Matrix aufgebaut. Deren Zeilen entsprechen den Knoten des einen Graphen und deren Spalten denen des anderen Graphen. Diese Matrix wird solange neu berechnet und normalisiert, bis sie konvergiert oder eine bestimmte Anzahl an Iterationen überschritten werden. Je höher ein Wert in der Matrix ist, desto wahrscheinlicher korrespondieren die beiden zu diesem Wert gehörenden Knoten. Danach wird noch eine Heuristik angewendet, um die Matrix zu bereinigen. Am Ende markieren die Einsen in der Matrix die Korrespondenzen, alle anderen Werte sind null. Die Autoren des Verfahrens beschreiben in [GrMat96] verschiedene durchgeführte Experimente mit diesem Algorithmus, unter anderem wurden zwei Arten von Tests durchgeführt:

- Graphen aus Bildern

  Die zur Testeingabe verwendeten Graphen wurden aus zweidimensionalen Bildern erzeugt, wobei die Knoten der Graphen markanten Punkten in den Bildern entsprechen. Diese Auswahl erfolgt entweder manuell durch einen Menschen oder automatisch durch bestimmte Filter und Verfahren zur Merkmalserkennungen.

- Zufallsgenerierte Graphen

  Hier wird zufallsbasiert ein Graph mit 100 Knoten erzeugt und durchgemischt. Daraufhin werden verschiedenartige Störungen hinzugefügt, wie z.B. das zufallsbasierte Entfernen von Kanten und das Ändern von Werten.

Für Graph Matching Algorithmen ist die Zeitkomplexität hier mit $O(k^2)$ relativ niedrig, wobei k die Anzahl der Kanten sowohl des einen als auch des anderen Graphen ist. Bei zwei vollständigen Graphen mit einer Knotenzahl von jeweils n ergeben sich jedoch jeweils n^2 viele Kanten, was zu einer Zeitkomplexität von $O(n^4)$ führt.

Point Matching

Das Point Matching Verfahren nach [PtMat98] ist eine Erweiterung des Graph Matching Verfahrens aus [GrMat96] und hat sehr ähnliche Eigenschaften wie dieses. Ziel ist hier auch die Korrespondenzfindung zwischen den Knoten zweier Graphen, jedoch sind hier diese Graphen aus Mengen von Punkten des 2D- und auch des 3D-Raumes entstanden. Der Algorithmus kommt auch mit vereinzelt fehlenden oder zusätzlich vorkommenden Punkten in den Mengen zurecht, wodurch jedoch der Aufwand zur Berechnung deutlich steigt.

Bei der Überführung der einen Menge in die andere werden Translation, Rotation, Skalierung und Scherung berücksichtigt. In [PtMat98] wird diese Tatsache mit Abbildung 21 veranschaulicht, hier werden sogar Korrespondenzen zwischen sehr unterschiedlichen Punktmengen handschriftlicher Ziffern gefunden.

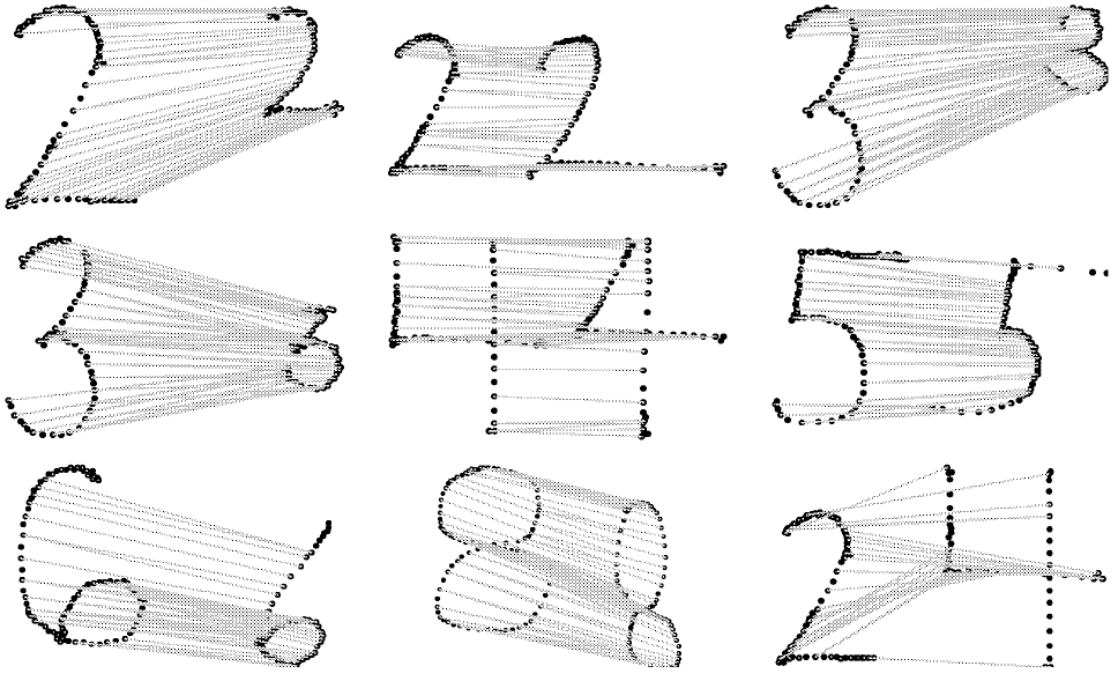


Abbildung 21: Korrespondenzen handgeschriebener Ziffern, Quelle: [PtMat98]

Skalierung und Scherung sind für die Korrespondenzfindung von Daten aus 3D-Laserscannern unnötig. Außerdem liegt trotz der beeindruckenden Ergebnisse ein großer Nachteil des Verfahrens darin, dass bei jedem Rechenschritt immer alle Kanten berücksichtigt werden, was in der Praxis zu unnötig vielen Berechnungen führt. Bei realen Scandaten gibt es meist nur wenige sehr ähnliche und viele unähnliche Muster, wobei der in dieser Arbeit vorgestellte Algorithmus die vielen unähnlichen ignoriert und damit im Durchschnitt eine deutlich bessere Performance hat.

7.5. Extrembeispiele

Hier werden einige Fälle betrachtet, in denen es dem Algorithmus nicht möglich ist, eine korrekte Lösung zu finden. Dafür kann es verschiedene Ursachen geben.

Mehrdeutige Lösungen

Werden Scans in verschiedenen Räumen gemacht, die die gleiche Geometrie haben und auch sonst exakt gleich aussehen, so ist eine automatische Unterscheidung zwischen diesen Räumen und somit auch eine Korrespondenzfindung prinzipiell nicht möglich. Das ist z.B. bei Scans in übereinanderliegenden Räumen unterschiedlicher Stockwerke eines Gebäudes der Fall.

Doch auch bei unterschiedlichen Geometrien kann eine Korrespondenzfindung prinzipiell unmöglich sein, z.B. wenn bestimmte Symmetrien Mehrdeutigkeiten erzeugen, wie etwa in Abbildung 22 dargestellt.

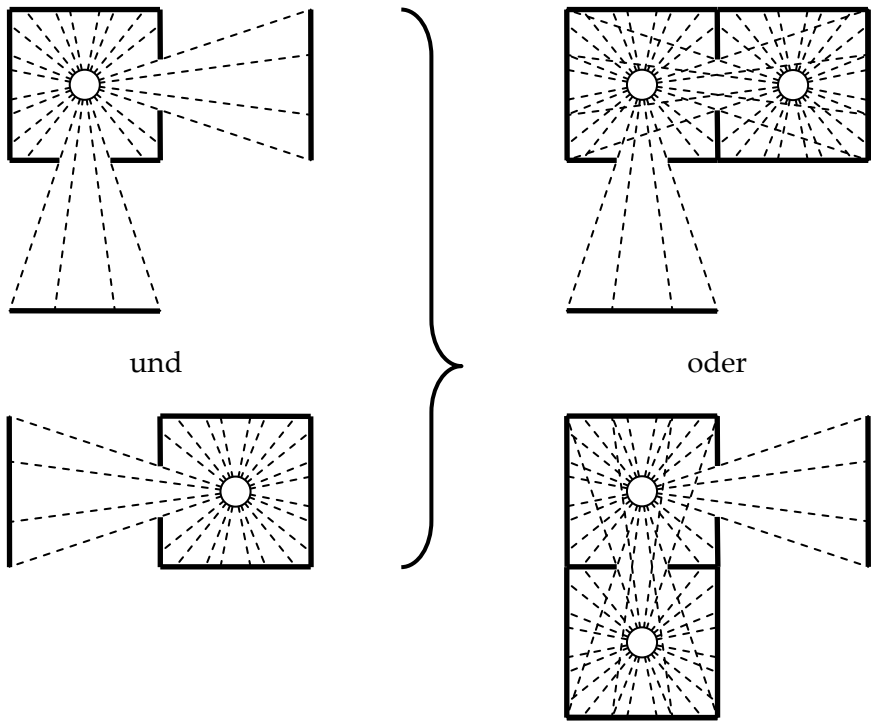


Abbildung 22: Mehrdeutige Registrierung

Wiederkehrende Muster

Es gibt Fälle, in denen wegen spezieller wiederkehrender Muster der vorgestellte Algorithmus (ohne Testregistrierung am Ende) falsche Korrespondenzen findet, obwohl es eine korrekte eindeutige Korrespondenz gibt. Es handelt sich dabei um theoretische Fälle, da es sehr schwierig sein dürfte, in der Realität die entsprechenden Relationsmuster zu erzeugen. Ein solches Beispiel ist in Abbildung 23 dargestellt.

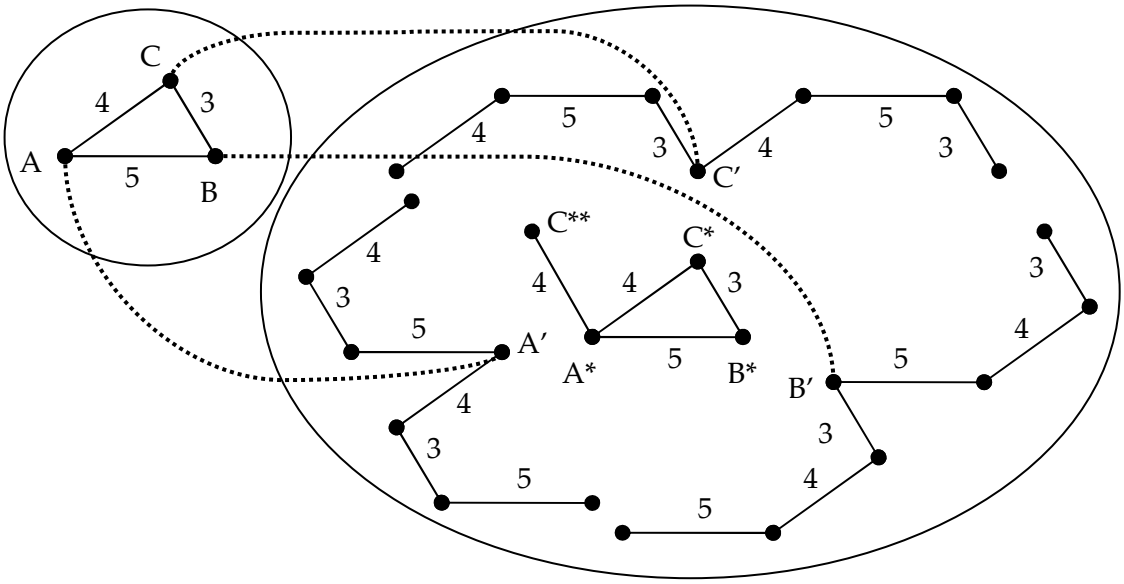


Abbildung 23: Falsche Korrespondenzfindung

Hierbei handelt es sich um zwei Graphen, die aus Lokal- und Referenzwolke entstanden sind. Alle Knoten entsprechen hier Punkttargets, alle Kanten bestehen aus den Targetabständen. Beide Graphen sind vollständig, allerdings sind zur besseren Übersicht alle unwichtigen Kanten des rechten Graphen nicht eingezeichnet. Diese können jeweils einen beliebigen, jedoch im linken Graphen nicht vorkommenden Wert haben. Es gibt genau eine maximale eindeutige Korrespondenz, nämlich (A,A*), (B,B*) und (C,C*), denn das entspricht dem größten gemeinsamen Teilgraphen. Wenn wir annehmen, dass nur drei Neuberechnungen der Ähnlichkeitsmatrix stattfinden, wird jedoch (A,A′), (B,B′) und (C,C′) fälschlicherweise als korrekte Korrespondenz berechnet. Dafür gibt es hauptsächlich zwei Gründe:

- Alle Pfade der Tiefe 3 von den „falschen“ Targets A′, B′ und C′ haben in der gleichen Reihenfolge die gleichen Werte wie die Pfade von den Lokaltargets.
- Das „richtige“ Target A* und damit auch B* und C* werden wegen der doppelt vorkommenden Relation 4 zu C** durch die Durchschnittsbildung geschwächt.

8. Implementierung

8.1. Objekteigenschaften

Jedes Target entspricht in einer möglichen Implementierung einem geometrischen Objekt mit gewissen Eigenschaften, wobei drei davon aus Benutzersicht besonders wichtig für die Korrespondenzfindung sind. Ihre englischen Bezeichnungen sind:

1. „Name“ (als Zeichenkette)
2. „Automatic Name“ (wahr oder falsch)
3. „Light“ (aus oder grün)

Jedes dieser drei Datenfelder wird persistent mit dem Objekt gespeichert und ist synchron zu seinem entsprechenden Oberflächenelement.

Abbildung 24: Eingabefeld für den Targetnamen

Einem Objekt kann auch ein Name vergeben werden, wenn die Anwendung keine automatische Korrespondenzfindung unterstützt. Unabhängig davon ist es sinnvoll, dass korrespondierende Objekte in verschiedenen Scans gleiche Namen erhalten.

Abbildung 25: Checkbox für „Automatic Name“

Der Benutzer soll selbst entscheiden, ob der Name eines Objektes durch das Programm automatisch verändert werden darf, falls eine Korrespondenz gefunden wird. Hierfür ist für jedes Objekt eine Checkbox implementiert. Wird diese gesetzt, so erhält Automatic Name den Wert „wahr“, andernfalls „falsch“. Standardmäßig ist der Wert auf „wahr“ gesetzt.

Abbildung 26: Grünes Licht

Um dem Benutzer eine Rückmeldung zu geben, ob der angezeigte Name erfolgreich durch eine Korrespondenzfindung gesetzt wurde, gibt es zusätzlich bei jedem Objekt ein Lichtsymbol. Dieses ist dann grün, wenn der Name dieses Objektes und der mindestens eines weiteren Objektes gleichen Namens jeweils durch eine gefundene Korrespondenz bestätigt wurden. Andernfalls ist das Licht aus.

8.2. Anwendungsfälle

Finde Korrespondenzen

Ein Anwendungsfall ist das Finden von Korrespondenzen für alle Objekte eines Scans. Dafür gibt es folgende Auslöser:

- Expliziter Aufruf „Find Correspondences“, z.B. im Kontextmenü eines Scans.
- Erkennung eines Objektes aus einem Marker (Pin im Bild) .
- Erkennung eines Objektes aus einem markierten Bereich.
- Setzen der „Automatic Name“-Checkbox von falsch auf wahr.

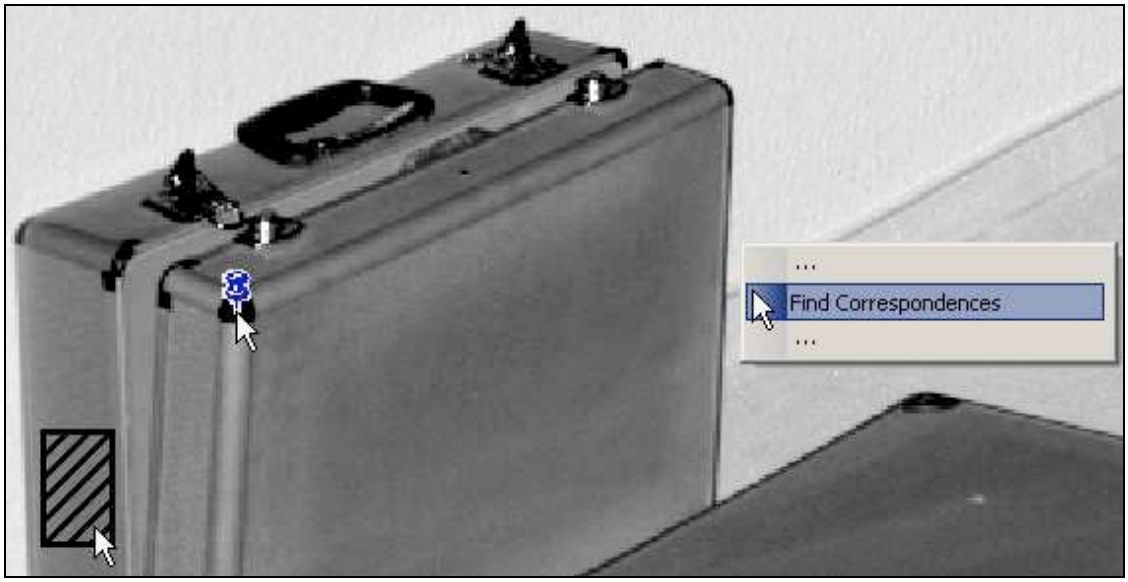


Abbildung 27: Auslöser der Korrespondenzfindung

Der beschriebene Algorithmus wird mehrfach ausgeführt, indem der aktuelle Scan paarweise mit jedem anderem Scan kombiniert wird, der im Strukturbaum ein direkter Nachbar auf derselben Ebene ist. Dabei werden nur Namen von Objekten des aktuellen Scans geändert, für die „Automatic Name“ wahr ist.

In den Einstellungen des Programms sollte es noch weitere Checkboxen geben, mit denen die Korrespondenzsuche deaktiviert werden kann. Damit hat der Benutzer die Möglichkeit, an einer zentralen Stelle alle Automatismen bezüglich der Namenswahl von Objekten zu deaktivieren. Diese Einstellung bezieht sich auch auf den im Folgenden beschriebenen Fall „Finde alle Korrespondenzen“, bei der die Cluster-Registrierung mit oder ohne Korrespondenzfindung stattfinden kann.

Finde alle Korrespondenzen

Eine gewöhnliche Registrierung findet zwischen zwei Scans statt. Hat man jedoch mehr als zwei unregistrierte Scans, so kann man mit einer Cluster-Registrierung alle auf einmal miteinander verknüpfen, ohne manuell die passenden Paarungen anzugeben. Meistens baut eine Cluster-Registrierung intern auf mehreren paarweisen Registrierungen auf, in diesem Fall kann der in dieser Arbeit vorgestellte Algorithmus jeweils vor jede dieser paarweisen Registrierungen ausgeführt werden.

Entfernen eines Objektes

Beim Entfernen eines Objektes aus einer Korrespondenz muss der „Light"-Status aller Objekte mit gleichem Namen aktualisiert werden. Dabei gibt es drei Fälle:

- Löschen eines einzelnen Objektes eines Scans.
- Löschen eines kompletten Scans mit all seinen Objekten.
- Deaktivieren einer „Automatic Name"-Checkbox.

Falls ein gelöschtes Objekt das letzte war, zu dem ein anderes Objekt eine Korrespondenz hatte, so muss der „Light"-Status dieses anderen Objektes auf aus gesetzt werden.

Umbenennen eines Objektes

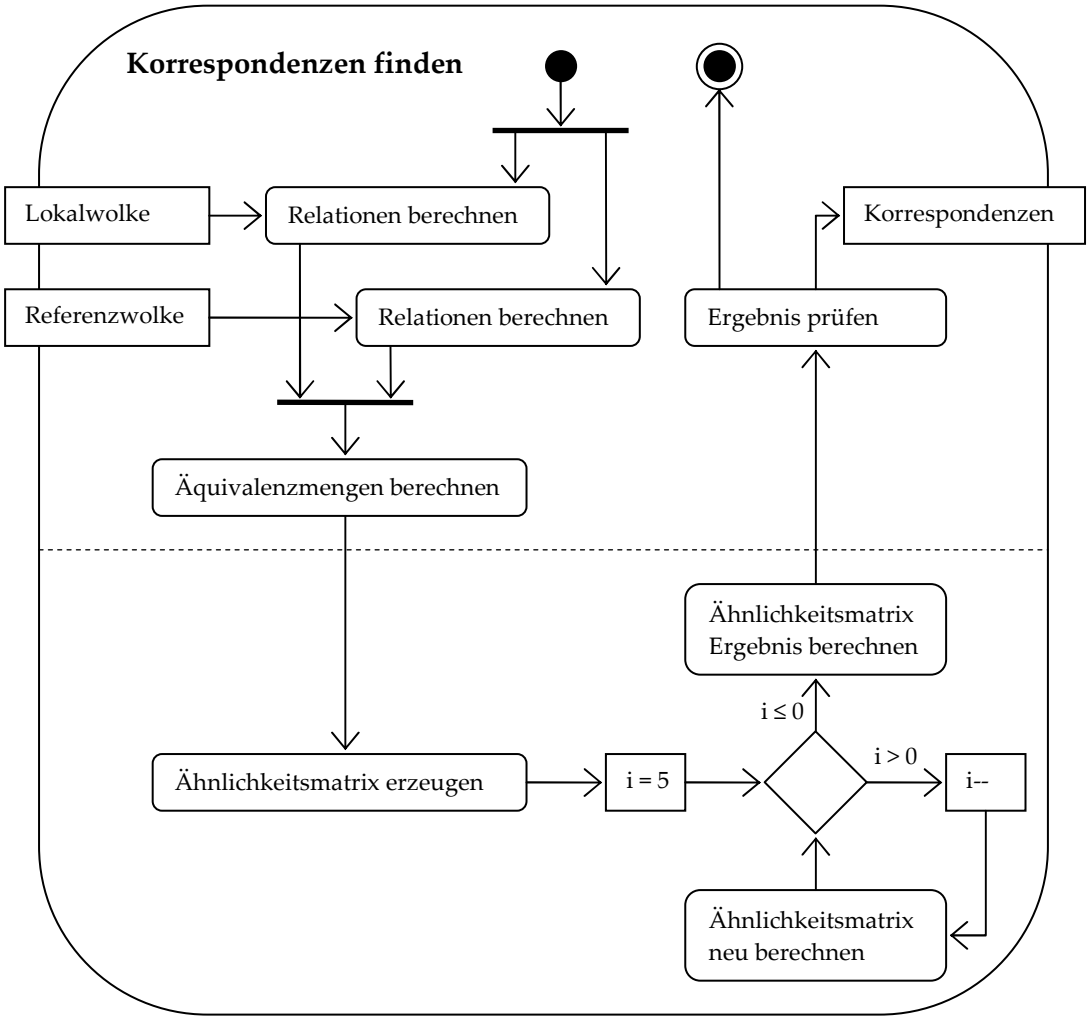
Der Name eines Objektes kann oft auf verschiedene Arten verändert werden, je nach Implementierung sind z.B. folgende denkbar:

- Änderung im Eingabefeld des Namens eines Objektes.
- Direktes Umbenennen in einem möglicherweise vorhandenen Strukturbaum.
- Umbenennung beim erstmaligen Erstellen eines Objektes, z.B. in einem durch eine Selektion mit einem Marker erschienenen Dialog.

Ist für das umbenannte Objekt und auch für alle dazu korrespondierenden Objekte „Automatic Name" wahr, so werden all diese Objekte entsprechend umbenannt. Das entspricht z.B. dem Fall, wenn der Benutzer das durch eine erfolgreich durchgeführte Korrespondenzfindung gefundene Target „Boden" für alle Scans auf einmal in „Teppich" umbenennen möchte.

Im anderen Fall, ist also „Automatic Name" des Objektes oder eines dazu korrespondierenden Objektes falsch, dann wird der „Light"-Status auf aus gesetzt und nur das aktuelle Objekt umbenannt. Der Benutzer ist also mit dem Ergebnis einer Korrespondenzfindung nicht einverstanden gewesen, woraufhin zur sicheren Seite keine automatische Umbenennung der korrespondierenden Objekte erfolgt.

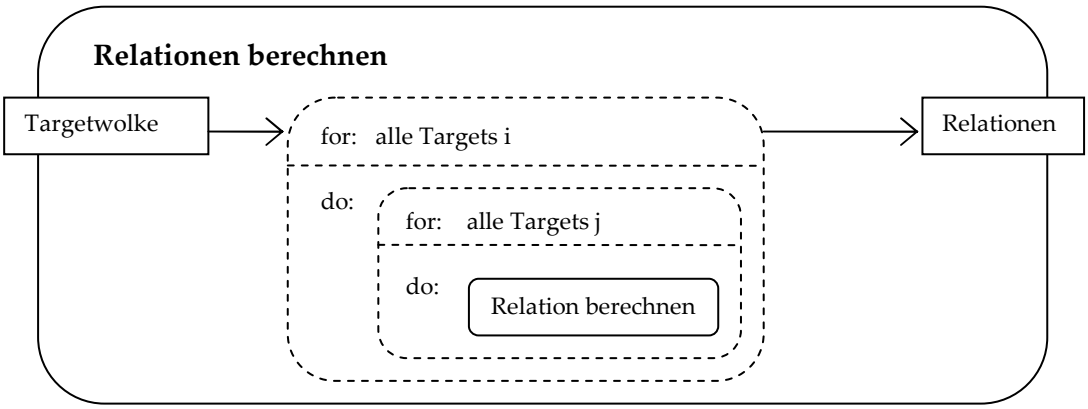
8.3. Korrespondenzfindung (Klasse CorrespondenceSearch)



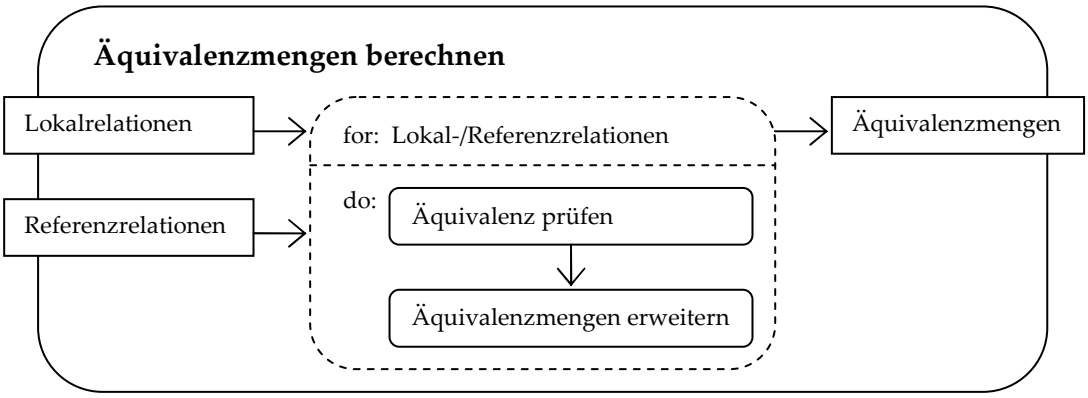
UML 1: Aktivität Korrespondenzen finden

Die Implementierung des Algorithmus ist prinzipiell in zwei Klassen aufgeteilt. „CorrespondenceSearch“ beinhaltet die Suche selbst mit allen in der Beschreibung des Algorithmus genannten Aufrufen. Die im nächsten Kapitel beschriebene Klasse „CorrespondenceTable“ kapselt dagegen die Ähnlichkeitsmatrix und alle zu ihr gehörenden Funktionen.

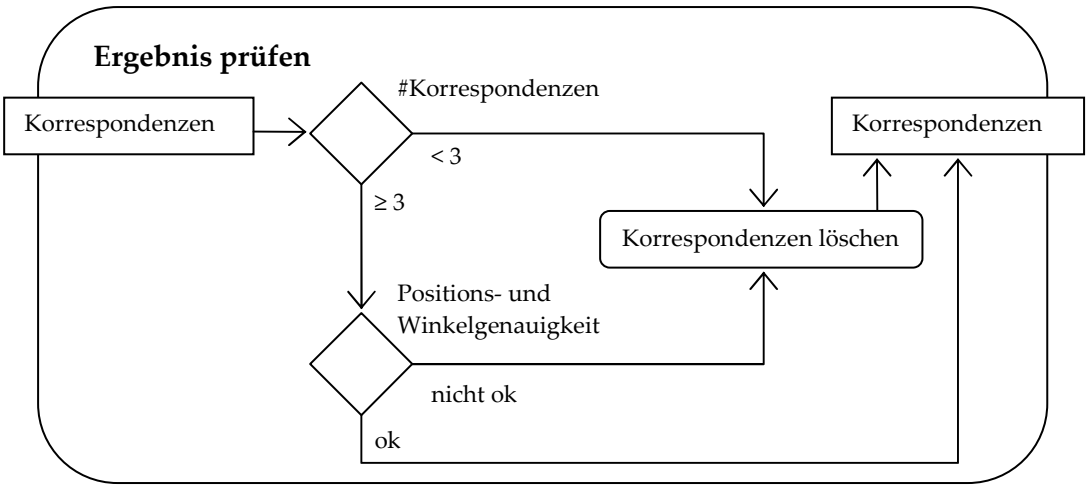
Aus den Mengen von Targets werden jeweils ihre Relationen aufgebaut. Als äquivalent geltende Relationen werden zusammengefasst und zur Erzeugung der Ähnlichkeitsmatrix verwendet. Diese wird mehrfach neu berechnet, wobei das beste Ergebnis geholt und eventuell noch einmal überprüft wird. Diese letzte Überprüfung kann entfallen, wenn sie bereits bei der Ähnlichkeitsmatrix erfolgt ist.



UML 2: Aktivität Relationen berechnen

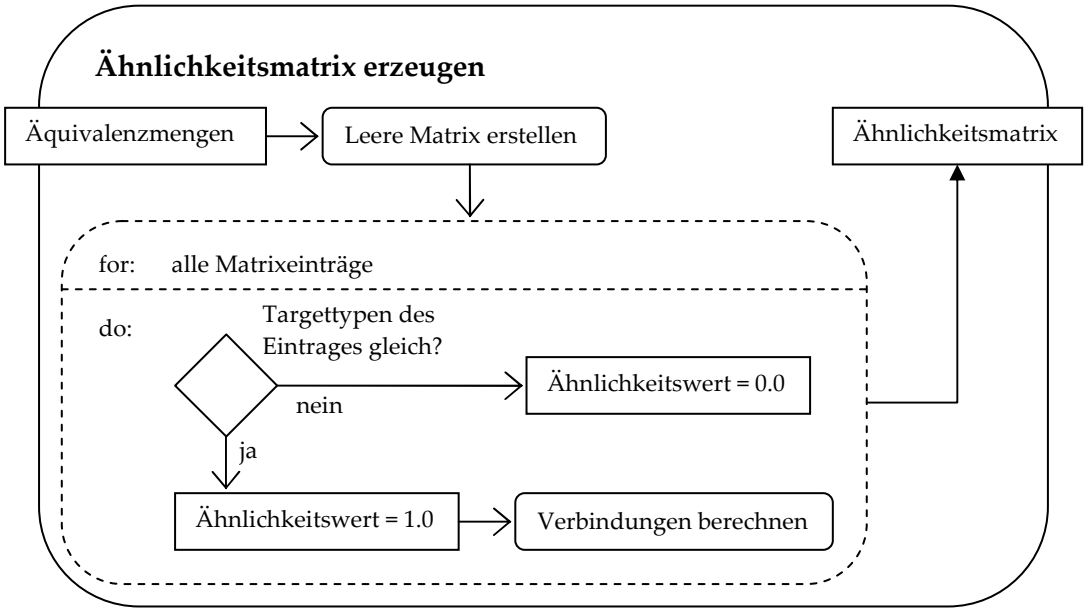


UML 3: Aktivität Äquivalenzmengen berechnen

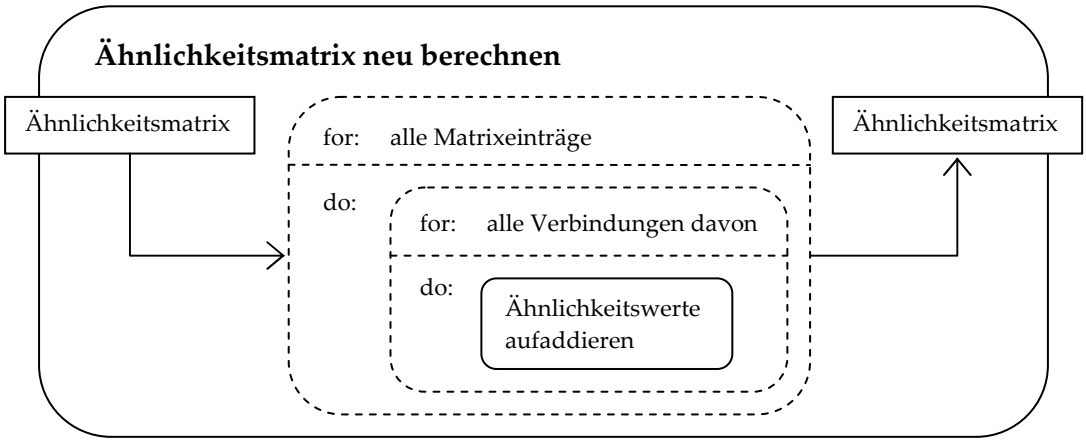


UML 4: Aktivität Ergebnis prüfen

8.4. Ähnlichkeitsmatrix (Klasse CorrespondenceTable)



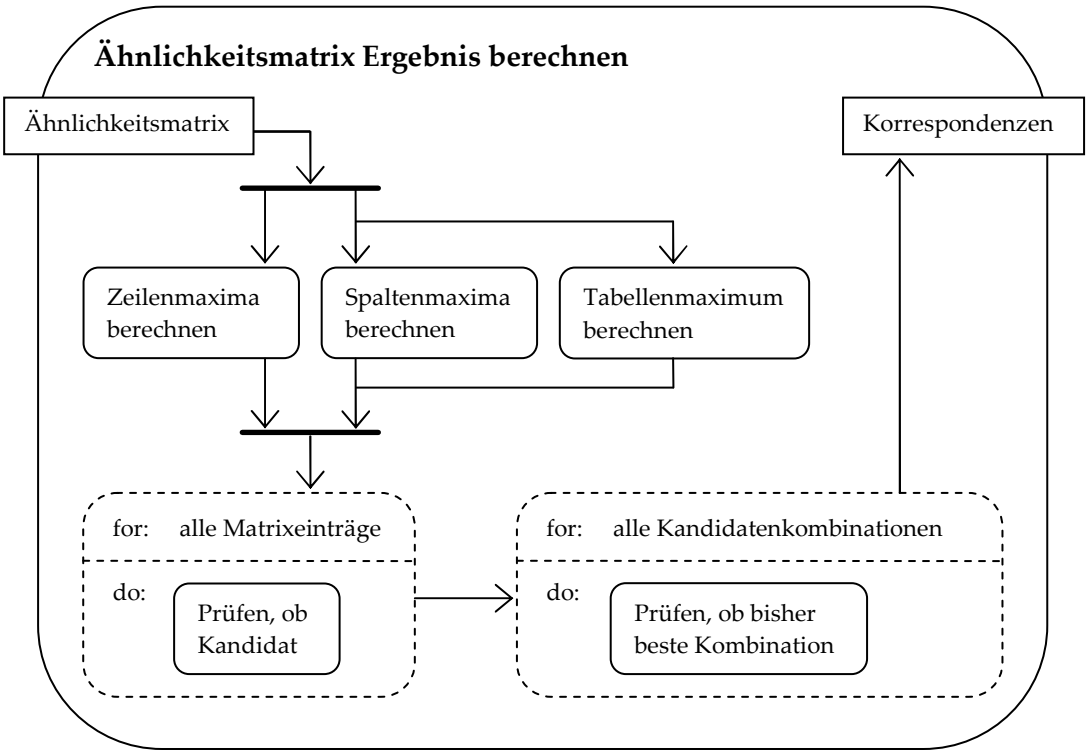
UML 5: Aktivität Ähnlichkeitsmatrix erzeugen



UML 6: Aktivität Ähnlichkeitsmatrix neu berechnen

Das Erzeugen der Ähnlichkeitsmatrix, ihre iterierte Neuberechnung und das Berechnen des Ergebnisses daraus sind mit der Klasse „CorrespondenceTable“ möglich. Die einzelnen Schritte entsprechen der Beschreibung des Algorithmus.

Bei der Berechnung der Verbindungen eines Matrixeintrages wird zu jeder Verbindung ein Gewicht gespeichert, welches von einem Target weggehende gleiche Relationen arithmetisch mittelt. Bei der iterierten Neuberechnung werden beim Aufaddieren der Ähnlichkeitswerte diese Gewichte berücksichtigt.



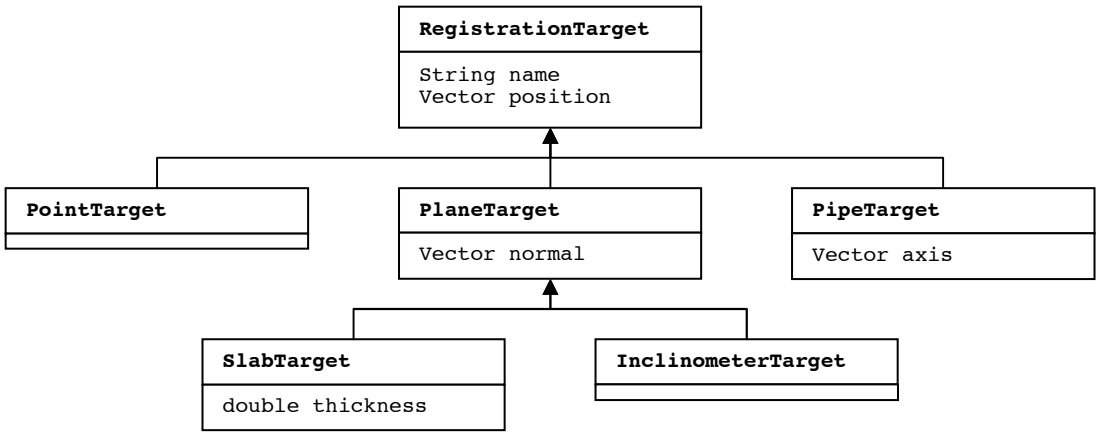
UML 7: Aktivität Ähnlichkeitsmatrix Ergebnis berechnen

Bei der Berechnung des Ergebnisses aus der Ähnlichkeitsmatrix wird für jeden Matrixeintrag geprüft, ob es groß genug im Verhältnis zu seinem Zeilenmaximum, Spaltenmaximum und Tabellenmaximum ist. Nur wenn das der Fall ist, handelt es sich bei dem Eintrag um einen Kandidaten. Wie bei einem Bit-Addierer, der jeweils um eins hochgezählt wird, werden alle Kandidatenkombinationen durchgegangen. Hierbei entspricht jede Zeile der Matrix einer Ziffer, die als mögliche Werte alle in dieser Zeile als Kandidat gekennzeichneten Einträge annehmen kann. Zusätzlich ist für jede Zeile auch keine Korrespondenz möglich. Für jede dieser Kombinationen wird eine Testregistrierung ausgeführt, die dann akzeptabel ist, wenn sie mindestens drei Korrespondenzen enthält und die Standardabweichungen der Winkel- und Positionsgenauigkeiten bei der Testregistrierung in Ordnung sind. Als neue beste Kombination gilt sie aber erst dann, wenn sie entweder mehr Korrespondenzen wie die bisher beste hat oder gleich viele Korrespondenzen, dafür aber ein besseres Registrierungsergebnis.

Jede der hier vorgestellten Aktivitäten ist in der Implementierung in einer eigenen Funktion gekapselt. Durch die schmalen Schnittstellen, auch zwischen den beiden Klassen „CorrespondenceSearch“ und „CorrespondenceTable“ ist ein nachträglicher Austausch einzelner Module problemlos möglich.

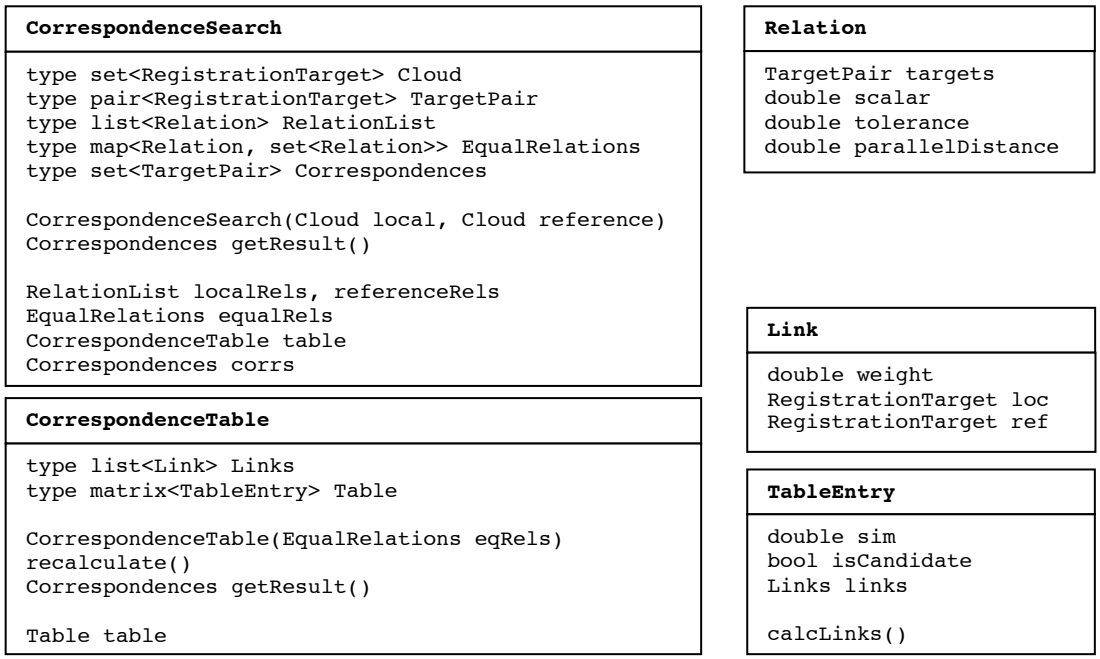
8.5. Klassenübersicht

Hier ist eine Übersicht der verschiedenen Klassen, die für die beschriebenen Targettypen verwendet werden. Eine Klasse hat zusätzlich zu ihren eigenen Datenelementen alle vererbten Elemente ihrer Elternklasse.



UML 8: Klassendiagramme – Targets

Die Klassendiagramme zur Korrespondenzsuche und zur Ähnlichkeitsmatrix, sowie alle darin integrierten Klassen sind hier mit ihren wichtigsten Datenelementen und Funktionen dargestellt. Die verwendete Syntax entspricht praktisch der von C++.



UML 9: Klassendiagramme – Korrespondenzen

8.6. Korrespondenzprüfung

Tests, welche die einzelnen Elemente der Implementierung überprüfen sollen, sind bereits in Kapitel 6 – Analysen beschrieben. Beachtet wurden hierbei die in [SE07] empfohlenen Vorangehensweisen für den Programmtest.

Außer diesen Unittests mit künstlichen Daten gibt es außerdem mit der Klasse „CorrespondenceCheck“ die Möglichkeit des Testens mit echten Scandaten. Dabei werden für eine Menge von Scans die Korrespondenzen überprüft, die durch die Namen ihrer Targets gebildet werden. Unterscheiden sich diese von den Korrespondenzen, die durch den Cluster-Algorithmus gefunden wurden, so erfolgt eine entsprechende Ausgabe. Bei diesem Vergleich können zwei Fälle auftreten:

- Der Algorithmus findet eine Korrespondenz, obwohl sie wegen verschiedener Namen der Targets keine Korrespondenz bilden. *Beispiel:* Der Algorithmus findet, dass das Target „Teppich“ im Scan „Wohnzimmer“ identisch ist mit dem Target „Boden“ im Scan „Küche“.
- Die Targets scheinen wegen gleicher Namen eine Korrespondenz zu bilden, der Algorithmus kann aber eine solche nicht finden. *Beispiel:* Der Algorithmus findet, dass das Target „WandNord“ des Scans „Wohnzimmer“ nicht identisch ist mit dem Target „WandNord“ des Scans „Küche“.

Intern wird das durch folgendes Vorgehen realisiert:

- Alle bestehenden Namen der Targets werden ausgelesen und gespeichert, genauso alle Korrespondenzen, die diese (alten) Namen bilden.
- Alle Targets werden umbenannt, so dass sie einen für alle Scans eindeutigen Namen haben. Das Target „WandNord“ des Scans „Küche“ wird dabei z.B. umbenannt in „WandNord_of_Küche“.
- Jetzt erfolgt die Cluster-Registrierung mit automatischer Korrespondenzsuche.
- Die neuen Namen der Targets werden gespeichert, genauso alle dadurch gebildete neuen Korrespondenzen.
- Alle Targets werden wieder umbenannt, so dass sie ihre alten Namen erhalten.
- In den alten Korrespondenzen erhalten die Targets die gespeicherten neuen Namen. Sind diese bei einer Korrespondenz unterschiedlich, dann trifft der erste der oben beschriebenen Fälle zu.
- In den neuen Korrespondenzen erhalten die Targets ihre gespeicherten alten Namen. Sind diese bei einer Korrespondenz unterschiedlich, stellt das den zweiten der oben beschriebenen Fälle dar.

9. Markante Punkte

9.1. Interest-Operatoren

Voraussetzung für eine erfolgreiche automatische Korrespondenzfindung und damit auch für eine Registrierung sind im Raum gefundene geometrische Objekte. Meistens sind das Punkttargets und Ebenentargets, beliebige andere Formen sind jedoch auch denkbar. Die automatische Erkennung der Targets in einem Scan ist der letzte Schritt hin zu einer vollautomatischen Registrierung. Dieses Kapitel behandelt die automatische Suche nach markanten Punkten im Raum, die insbesondere für den Algorithmus zur Korrespondenzfindung als Eingabe geeignet sind. Ein markanter Punkt, oft auch „Point of Interest“ genannt, ist z.B. eine Raumecke. In [Corr86] werden Anforderungen an gute markante Punkte genannt. Übertragen auf Scandaten im 3D-Raum sind die wichtigsten Merkmale daraus:

- Alleinstellung: Der Punkt hat Eigenschaften, die ihn von anderen Punkten in seiner Umgebung unterscheiden, z.B. ist er deutlich heller oder dunkler.
- Wiedererkennung: Punkte, die von einem Verfahren als markant eingestuft werden, sollten ihn erneut als markant einstufen, wenn der Scan im Raum verschoben, gedreht oder sogar verzerrt wird.
- Stabilität: Kleinere Fehler und Störungen in den Scandaten sollten keine große Auswirkung auf das Finden der markanten Punkte haben.
- Seltenheit: In seiner Umgebung sollte ein markanter Punkt dieser Art möglichst selten sein. Wenn z.B. auf einer Fläche ein regelmäßiges Punktmuster vorliegt, dann sollen nicht alle Punkte des Musters als markant eingestuft werden.

Operatoren, die markante Punkte finden, werden auch als Interest-Operatoren bezeichnet. Auf zwei wird in dieser Arbeit besonders eingegangen, da sie in verschiedenen Situationen gute Ergebnisse liefern:

- FAST-Operator aus [Fast05].
- Förstner-Operator aus [Foer87].

Viele Laserscanner liefern außer den Punktwolken auch zweidimensionale Graustufenbilder des Scans, wobei der Grauwert eines Punktes der Reflektion des Laserstrahls entspricht. Dieses Bild wird in einem ersten Schritt für die Anwendung der Operatoren verwendet. Erst in weiteren Schritten werden auch die Abstände der Punkte vom Laserscanner berücksichtigt, also die Tiefeninformation des Bildes. Punkte, die von beiden Operatoren gefunden wurden und welche zusätzlich alle weiteren Kriterien erfüllen, kommen in die endgültige Wahl der markanten Punkte.

9.2. FAST-Operator

Der FAST-Operator ist eine in [Fast05] beschriebene Methode zur Erkennung markanter Punkte, z.B. in Graustufenbildern. „FAST“ bedeutet „Features from Accelerated Segment Test“ und ist zugleich eine Anspielung auf die sehr hohe Geschwindigkeit des Verfahrens.

Um festzustellen, ob ein bestimmter Punkt markant ist oder nicht, wird um den Punkt eine kreisförmige Maske gelegt. Pixel sind diskrete Einheiten, deshalb wird ein Bresenham-Kreis verwendet, welcher eine Diskretisierung der kontinuierlichen Kreislinie auf einem Raster darstellt. Der Radius des Kreises beträgt drei Pixel.

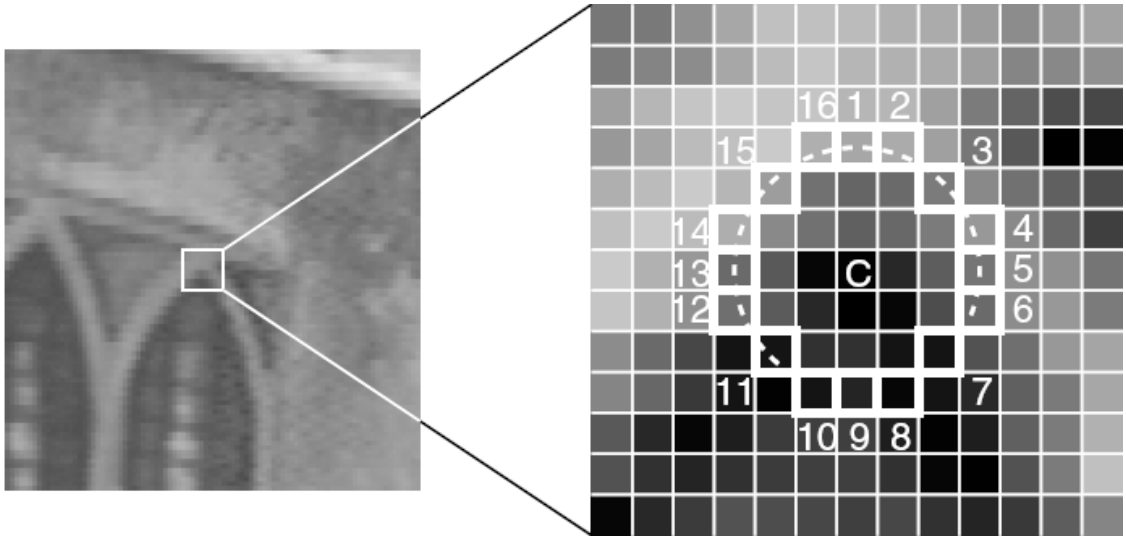


Abbildung 28: Fast-Operator, Quelle: [Fast05]

Der Mittelpunkt des Kreises gilt dann als ein markanter Punkt, wenn ein ausreichend großer zusammenhängender Bereich der Kreismaske existiert, bei dem alle Pixel entweder deutlich heller oder deutlich dunkler als der Mittelpunkt sind. Die Mindestgröße dieses Bereiches beträgt oft ¾ des Kreises, hier also 12 Pixel.

Beispiel

In der Abbildung ist C ein markanter Punkt, weil es auf der kreisförmigen Maske um C einen zusammenhängenden Bereich von mindestens 12 Pixeln gibt (Pixel 11 bis 16 und 1 bis 6), bei dem jeder einzelne dieser Punkte um einen festgelegten Betrag heller ist als C.

Zahlreiche Optimierungen existieren, z.B. können zunächst die vier Pixel 1, 5, 9 und 13 der Hauptrichtungen überprüft werden. Bei einem markanten Punkt in C müssen mindestens drei von ihnen sich deutlich vom Mittelpunkt unterscheiden.

9.3. Förstner-Operator

Der in [Foer87] beschriebene Förstner-Operator verwendet eine quadratische Fläche als Maske, hier im Beispiel von der Größe 3x3 Pixel. Aus dieser Maske werden alle Ableitungen g_y in y-Richtung und alle Ableitungen g_x in x-Richtung berechnet.

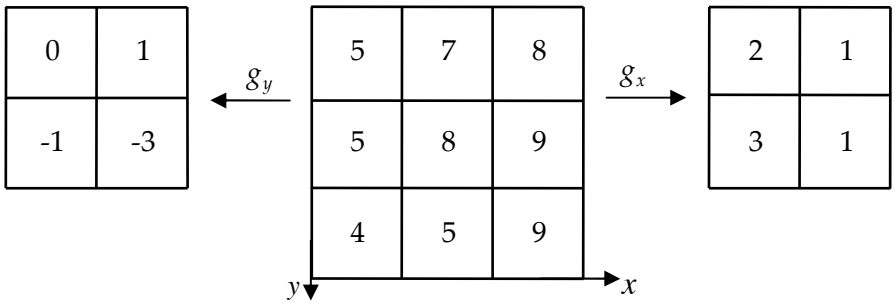


Abbildung 29: Förstner-Operator

Aus den Ableitungen wird die Autokorrelationsmatrix A aufgebaut, aus welcher die Parameter w und q berechnet werden können:

$$A = \begin{pmatrix} \sum g_x^{\,2} & \sum g_x g_y \\ \sum g_x g_y & \sum g_y^{\,2} \end{pmatrix},\ w = \frac{\det A}{\text{spur } A},\ q = \frac{4 \cdot \det A}{(\text{spur } A)^2}$$

Von allen Punkten, bei denen q einen Schwellwert von 0.8 überschreitet, werden diejenigen 50% mit dem größeren w ausgewählt. Diese Punkte gelten als markant. Gibt es in der Maske eine Kante in x-Richtung (waagerecht), so sind die Beträge der Ableitungen g_y in der Summe groß. Für Kanten in y-Richtung gilt das analog für g_x.

Beispiel

In der Abbildung sind links und rechts von der Maske die Ableitungen in y- und x-Richtung, deren Werte jeweils der Differenz der Grauwerte entsprechen. Dann ist:

$$\sum g_y^{\,2} = 0^2 + 1^2 + (-1)^2 + (-3)^2 = 11,\ \sum g_x^{\,2} = 2^2 + 1^2 + 3^2 + 1^2 = 15,\ \sum g_x g_y = -5,$$

$$w = \frac{15 \cdot 11 + (-5) \cdot (-5)}{15 + 11} \approx 7.3,\ q = \frac{4 \cdot (15 \cdot 11 + (-5) \cdot (-5))}{(15 + 11)^2} \approx 1.1 > 0.8.$$

Wenn wir annehmen, dass 7.3 größer ist als der Median aller w, für die $q > 0.8$ ist, dann liegt hier also ein markanter Punkt vor.

Je deutlicher sich eine Kante im Gradientenbild von der anderen unterscheidet, desto niedriger ist q und desto unwahrscheinlicher ist es, dass der Punkt markant ist. Bei zwei perfekt rechtwinkligen kontrastreichen Linien ist w sehr groß, der Punkt ist wahrscheinlich markant.

9.4. Optimierungen

Die beiden beschriebenen Operatoren liefern brauchbare Ergebnisse. In der praktischen Anwendung werden jedoch sehr viele Punkte gefunden, die nicht wirklich markant sind. Diese unerwünschten Effekte können vermieden werden, indem nach bestimmten Regeln gefundene markante Punkte wieder gelöscht werden.

Gleichverteilung

Markante Punkte sollten in etwa gleichmäßig über den gesamten Scan verteilt sein. Vor allem für die Korrespondenzfindung nützt es nicht viel, wenn zwar viele gute Punkte gefunden werden, diese sich aber alle in einem sehr kleinen Bereich befinden. Um dem entgegenzuwirken, wird als Optimierung das Bild in einzelne Zellen aufgeteilt. Für jede einzelne Zelle wird dann mit eigenen lokalen Kriterien nach den besten markante Punkten in dieser Zelle gesucht.

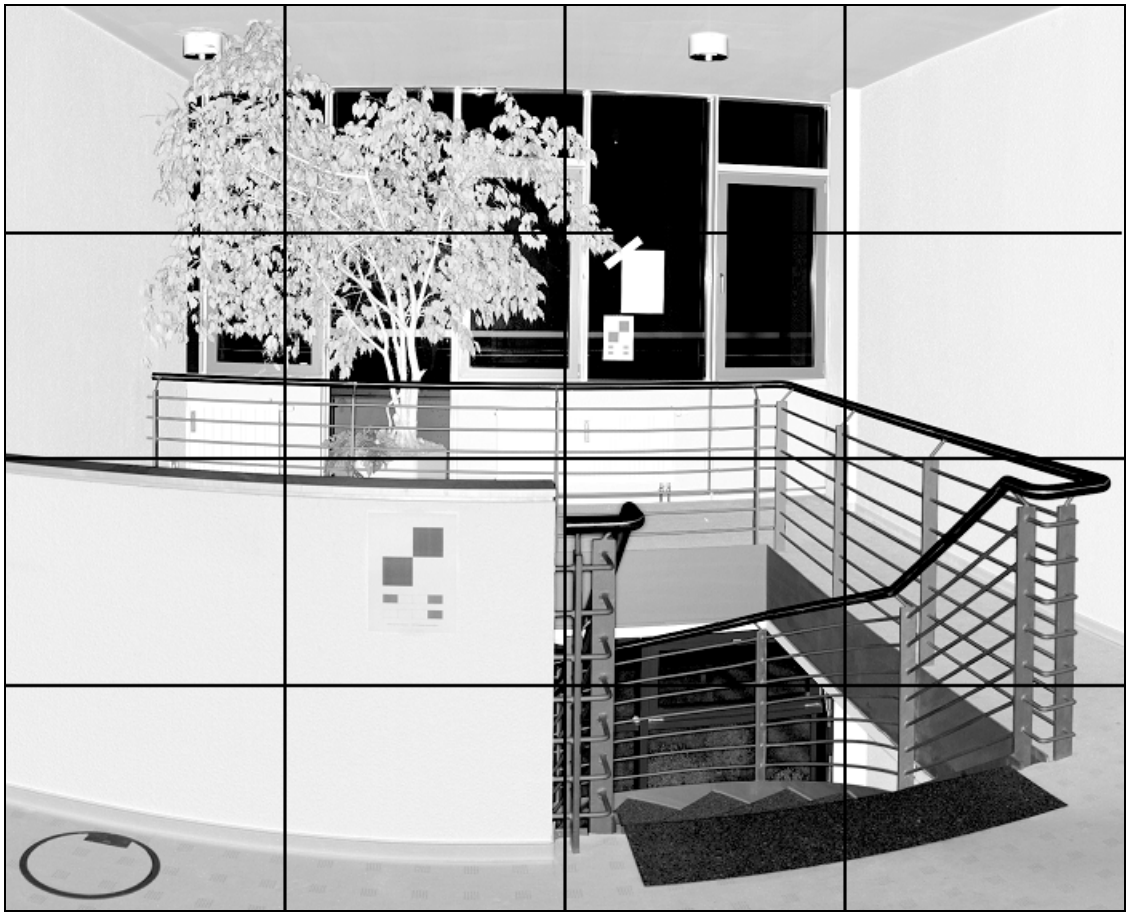


Abbildung 30: Zelleinteilung von Scans

Musterpunkte

Oft treten bestimmte Muster auf, wie z.B. gepunktete Flächen, bei denen die Operatoren zu viele markante Punkte finden. Die Lösung besteht darin, bei zu vielen Punkten mit sehr ähnlichem Wert von Interesse in einer Zelle, die auch noch in etwa gleichmäßig darin verteilt sind, diese nicht als markant einzustufen. Hier in der Abbildung werden fast alle Kreuzungspunkte der Fugen gefunden. Für eine Korrespondenzsuche ist diese unüberschaubare Anzahl störend.

Abbildung 31: Muster in Scans

Eine optimierende Erweiterung dazu ist, sich dieses Muster in einer bestimmten Form zu merken. So könnte hier z.B. zur Ebene Boden ein Zusatzparameter berechnet werden, der dieses regelmäßige Muster beschreibt. Dieser Parameter kann z.B. definiert werden als der durchschnittliche Abstand zweier zunächst als markant eingestufter Punkte dieser Ebene, hier entspricht das in etwa der Breite einer Kachel. Diese Zusatzinformation stellt für die Korrespondenzfindung eine große Hilfe dar.

Identische Punkte

Oft werden mehrere Punkte gefunden, die eigentlich dem gleichen markanten Punkt entsprechen. Die Lösung ist ein Löschen aller Punkte, zu denen in unmittelbarer Nähe weitere Punkte gefunden wurden.

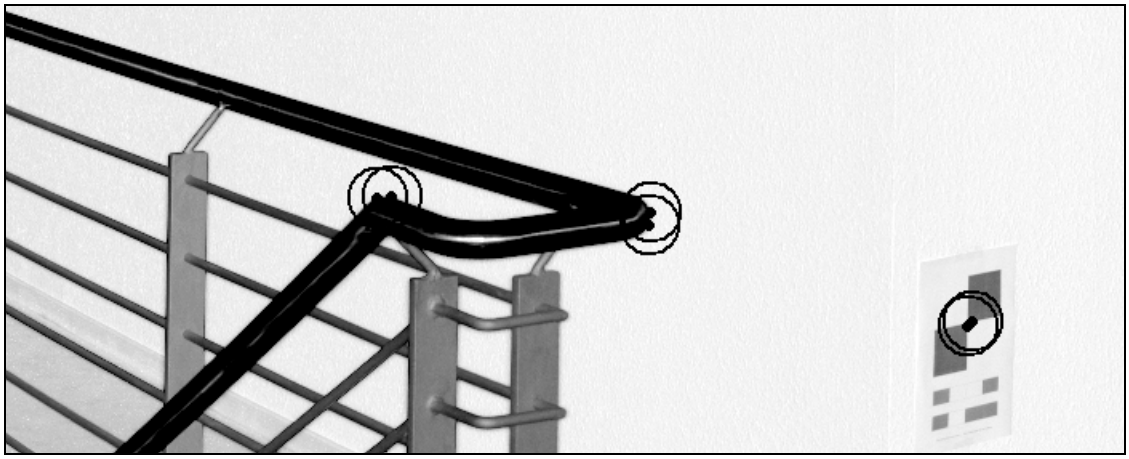


Abbildung 32: Identische Punkte in Scans

Kreuzungspunkte

Optisch kreuzen sich zwei Linien und bilden scheinbar einen markanten Punkt, obwohl die eine Linie deutlich vor der anderen liegt. Bei einer Veränderung der Scannerposition würde sich der Punkt ändern oder verschwinden. Die Lösung ist, dass alle Punkte der Maske in etwa dieselbe Tiefe haben müssen, damit der Punkt markant ist. Die Entfernung zum Scanner darf also nicht zu sehr springen.

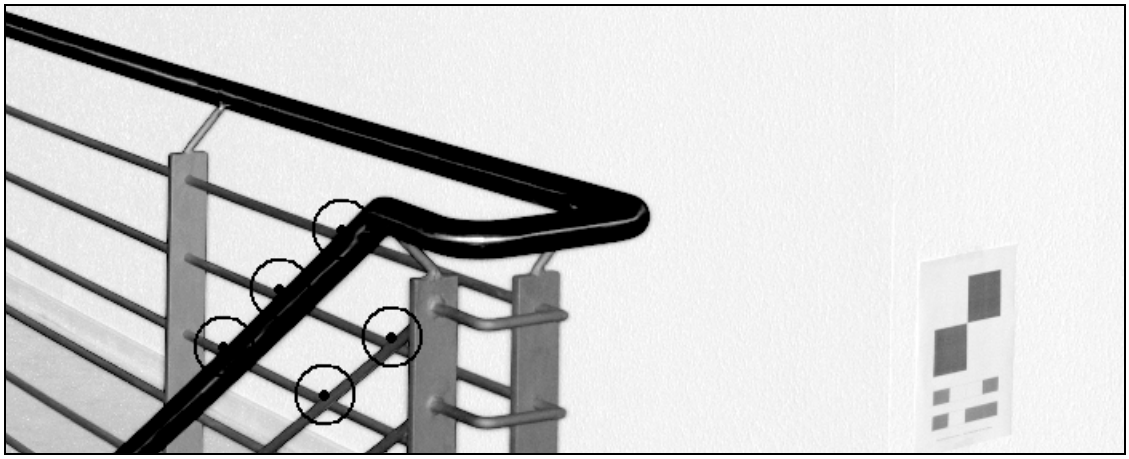


Abbildung 33: Optische Kreuzungspunkte in Scans

Schattenlinien

Kreuzt eine Schattenlinie eine Kante, oder trifft die Ecke eines Schattens auf eine Fläche, dann werden diese Punkte oft als markant eingestuft. Das Problem ist ähnlich zu dem der Kreuzungspunkte, jedoch gibt es hier nicht ohne weiteres eine Lösung. Im Allgemeinen in es nicht möglich zu wissen, ob eine verdunkelte Linie ein unerwünschter Schatten ist, oder ob sie z.B. einer tatsächlich dunkleren Färbung der Wand entspricht.

Hier im Beispiel sind die gefundenen Punkte durchaus markant. Würde man allerdings einen Mechanismus einbauen, der bei dunkleren Flächen auf Ebenen von Schatten ausgeht, dann würden auch die Ecken der Quadrate des Schachbretttargets als nicht markant eingestuft werden.

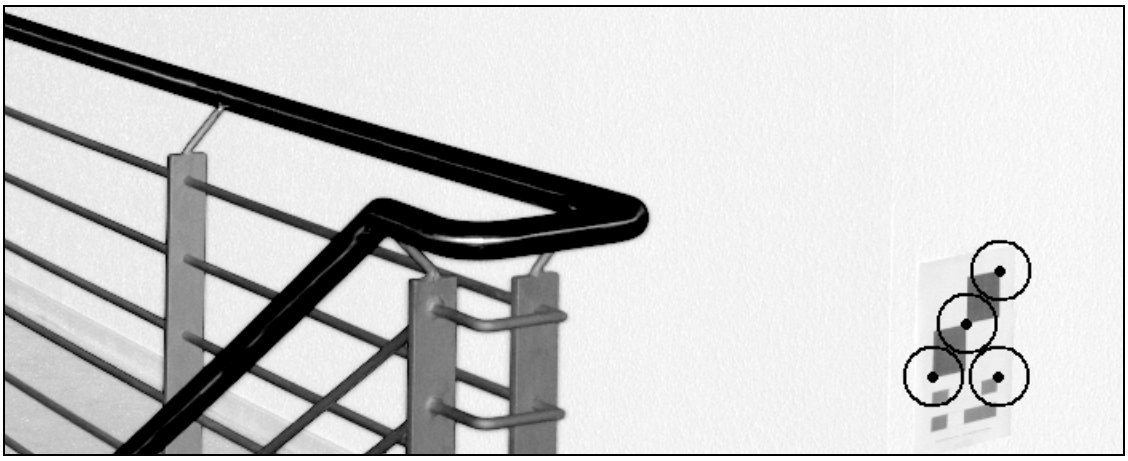


Abbildung 34: Scheinbare Schattenpunkte in Scans

Globale Selektion

Es kann sein, dass ein Punkt in einer Zelle sehr markant ist, auf das Gesamtbild bezogen jedoch überhaupt nicht. Das ist dann der Fall, wenn eine Zelle nichts Interessantes enthält, wie z.B. in Abbildung 30 die erste Zelle in der dritten Zeile. Die Lösung ist eine globale Selektion, die von allen gefundenen markanten Punkten des ganzen Bildes die schlechtesten 10% löscht. Als Vergleichswert dient dabei das von den Operatoren berechnete Maß an Interesse für einen Punkt.

Eine weitere Möglichkeit zur genaueren Differenzierung besteht darin, von beiden Operatoren gefundenen Punkten einen höheren Wert an Interesse zu geben. Genauso können Punkte bevorzugt werden, in deren unmittelbarer Umgebung weitere Punkte waren, diese jedoch durch die Auswahl identischer Punkte herausgefiltert wurden. Der Zeitpunkt dieser globalen Selektion sollte eher gegen Ende der Optimierungen stattfinden, damit andere herausgefilterte Punkte nicht in die Berechnung einfließen.

9.5. Maskenskalierung

Sowohl der FAST-, als auch der Förstner-Operator finden bei gleichen Scans in verschiedenen Auflösungen unterschiedliche markante Punkte. Der Grund liegt an der konstanten Größe der Maske in Pixeln. Bei einer hohen Auflösung ist der Ausschnitt der Maske entsprechend kleiner, was im Durchschnitt zu kleineren Gradienten in den Grauwerten und damit zu einer anderen Bewertung des Punktes führt. Es gibt mehrere Methoden, diesen unerwünschten Effekt zu beseitigen:

- 2D-Ausdünnen des Scans: Es werden gleichmäßig viele Pixel entfernt, bis die Anzahl der Pixel pro Winkel einen festgelegten konstanten Wert erreicht hat.
- 3D-Ausdünnen des Scans: Je weiter ein Punkt von dem Scanner entfernt ist, desto weniger stark wird seine Umgebung ausgedünnt. Damit kann man erreichen, dass der Flächeninhalt des durch die Maske abgedeckten Bereichs in etwa konstant bleibt. Vom Scanner unterschiedlich weit entfernte Punkte, die ähnlich markant sind, erhalten so ähnlichere Interessen-Bewertungen.

Fast den gleichen Effekt erhält man, indem stattdessen die Maske gestreckt wird, wobei die Anzahl der Pixel der Maske jedoch konstant bleibt. In Abbildung 35 ist eine doppelte und dreifache Streckung zweier Masken veranschaulicht.

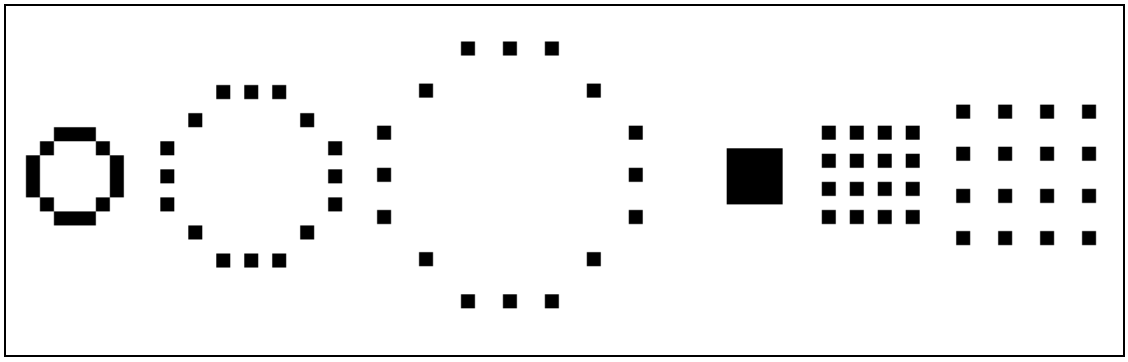


Abbildung 35: Skalierung von Masken

Statt der alleinigen Verwendung einer gestreckten Maske kann man auch für denselben Punkt mehrere Maskengrößen anwenden und die einzelnen Bewertungen miteinander vergleichen. Erst wenn für einen Punkt bei beiden Operatoren für alle verwendeten Maskengrößen hohe Werte vorliegen, liegt ein markanter Punkt vor.

Größere Masken haben den Vorteil, Schnittpunkte unscharfer Kanten besser zu erkennen. Außerdem haben große Masken eine nicht ganz so lokale Sicht das Bild und finden somit z.B. Ecken eines Raumes eher als Ecken auf sehr kleinen Objekten. Besonders für die Korrespondenzfindung haben solche globaleren markanten Punkte einen großen Nutzen.

9.6. Kombination

Kombiniert man einige der genannten Optimierungen mit beiden Operatoren und fügt noch eine Berücksichtigung der Tiefeninformation hinzu, die über die bei den Kreuzungspunkten beschriebene hinausgeht, dann ist Abbildung 36 ein mögliches Ergebnis der Suche. Viele dieser Punkte eignen sich hervorragend als Eingabe für den Algorithmus zur Korrespondenzfindung. Trotz aller Optimierungen gibt es aber noch viele Punkte, die von einer anderen Perspektive aus nicht mehr sichtbar sind oder von dieser aus nicht mehr als markant eingestuft werden.

Meistens bringen Punkte mehr Informationen in eine Registrierung und Korrespondenzfindung mit ein als Ebenen. Dennoch wäre es vielversprechend, einen Algorithmus zur automatischen Erkennung großer Ebenen einzusetzen, da diese deutlich robuster sind in Bezug auf Änderungen in der Perspektive des Scanners. Um die bei Ebenen oft vorkommenden Mehrdeutigkeiten aufzulösen, könnten wenige sehr markante Punkte dann als Symmetriebrecher dienen.

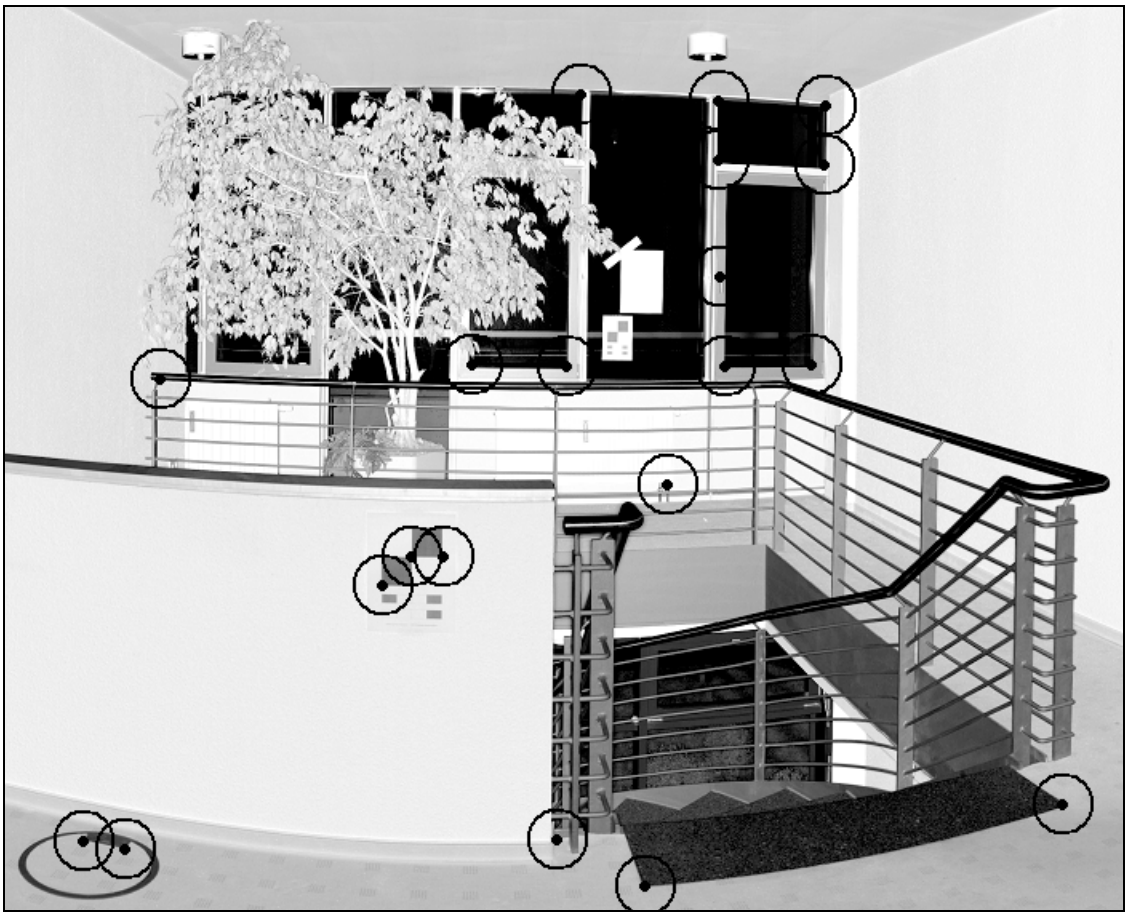


Abbildung 36: Auswahl gefilterter markanter Punkte

10. Ausblick

Für den Anwender einer Scansoftware ist es wichtig, ob die einzelnen Schritte einer Registrierung vollautomatisch durch die Software ausgeführt werden können, oder ob ein großer manueller Aufwand dafür nötig ist. Eine mögliche Einteilung des Grades der Automatisierung für die Registrierung von Scandaten ist folgende:

Grad		Selektion oder Erkennung von Objekten	Korrespondenzfindung	Positionierung aus den Korrespondenzen	Positionierung direkt aus den Scandaten
1.	Mensch Software				**X**
2.	Mensch Software	**X**	**X**	**X**	
3.	Mensch Software	**X**	**X**	**X**	
4.	Mensch Software	**X**	**X**	**X**	
5.	Mensch Software				**X**

Tabelle 18: Grad der Automatisierung

Beim 1. Grad muss der Anwender die Scandaten solange manuell drehen und verschieben, bis sie zusammenpassen. Abgesehen vom hohen Aufwand sind dabei präzise Ausrichtungen kaum möglich. In Richtung einer vollständig automatisierten Registrierung ermöglicht der in dieser Arbeit vorgestellte Algorithmus den Schritt vom 2. zum 3. Grad. Zusammen mit der Erkennung markanter Punkte im Raum und eventuell einem Verfahren zur Erkennung von Ebenen ist der Schritt zum 4. Grad keine große Hürde mehr.

Deutlich schwieriger wird es bei einer automatischen Lösung, wenn man auf die Betrachtung einzelner Objekte verzichtet, womit das Anbringen künstlicher Targets nicht mehr nötig wäre. Der 5. Grad stellt eine solche ganzheitliche Lösung dar, bei der die Scandaten direkt verglichen werden und ihr bestes Zusammenpassen automatisch berechnet wird. Im Grunde entspricht das der manuellen Lösung vom 1.Grad, nur das diese automatisch von der Software ausgeführt wird.

Das Problem ist dabei sowohl die große Menge an Daten, als auch die Tatsache, dass Software in absehbarer Zeit nicht das nötige Hintergrundwissen haben wird, um im Kontext wirklich zu verstehen, welche Inhalte die Scans haben und wie sie in der Realität korrekt zusammenpassen.

Literatur

[SE07] Ludewig, Jochen; Lichter, Horst: Software Engineering – Grundlagen, Menschen, Prozesse, Techniken. dpunkt.verlag, Heidelberg, 2007.

[Faro08] FARO EUROPE GmbH & Co. KG, 2008.

[Korr05] Haase, Kristine: Semi-automatische Korrespondenzfindung und Registrierung von Lebergefäßmodellen. Universität Koblenz-Landau, 2005.

[Math04] Casselman, Bill: Mathematical Illustrations, Chapter 12: Transformations in 3D. Cambridge University Press, 2004.

[DIN9300] DIN 9300: Begriffe, Größen und Formelzeichen der Flugmechanik, Teil 1: Bewegung des Luftfahrzeugs gegenüber der Luft.

[AuSrc98] Kleinberg, Jon M.: Authoritative Sources in a Hyperlinked Environment. Proc. 9th ACM-SIAM Symposium on Discrete Algorithms, 1998.

[Page99] Page, Lawrence; Brin, Sergey; Motwani, R.; Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Stanford University, 1999.

[IR08] Manning, Christopher D.; Raghavan, Prabhakar; Schütze, Hinrich: Introduction to Information Retrieval. Cambridge University Press, 2008.

[Matrix96] Golub, Gene H.; Van Loan, Charles F.: Matrix Computations. The Johns Hopkins University Press, Baltimore, 1996.

[GrMat96] Gold, Steven; Rangarajan, Anand: A Graduated Assignment Algorithm for Graph Matching. IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 18, 1996.

[PtMat98] Gold, Steven; Rangarajan, Anand; Lu, Chien-Ping; Pappu, Suguna; Mjolsness, Eric: New Algorithms for 2D and 3D Point Matching: Pose Estimation and Correspondence. Pattern Recognition, Volume 31, 1998.

[Corr86] Förstner, Wolfgang: A Feature Based Correspondence Algorithm for Image Matching. International Archives of Photogrammetry and Remote Sensing, Volume 26, 1986.

[Fast05] Rosten, Edward; Drummond, Tom: Fusing Points and Lines for High Performance Tracking. Tenth IEEE Conference on Computer Vision, 2005.

[Foer87] Förstner, Wolfgang; Gülch, Eberhard: A Fast Operator for Detection and Precise Location of Distinct Points, Corners and Centres of Circular Features. ISPRS Intercommission Workshop, 1987.

Verzeichnis

Abbildungen

Abbildung 1: 3D-Laserscanner, Quelle: [Faro08] ... 9
Abbildung 2: Registrierung – Ablauf ... 10
Abbildung 3: Registrierung – Beispiel ... 10
Abbildung 4: Dreidimensionalität der Bilddaten ... 13
Abbildung 5: Parameter der Transformation ... 14
Abbildung 6: Ablauf einer Korrespondenzfindung ... 17
Abbildung 7: Ähnlichkeitssuche bei fehlenden Targetkombinationen ... 19
Abbildung 8: Parallelabstand ... 22
Abbildung 9: Hubs- and Authorities Berechnung ... 24
Abbildung 10: Beispiel zu Precision und Recall ... 25
Abbildung 11: Lokal- und Referenzwolke als Eingabe ... 27
Abbildung 12: Relationen als Kanten eines Graphen ... 29
Abbildung 13: Registrierung ... 35
Abbildung 14: Beispiel zur Variante mit Gewichtungen ... 37
Abbildung 15: Beispielberechnung der Metriken ... 43
Abbildung 16: Beispiel zu LINE_MAX_THRESH und TOTAL_MAX_THRESH ... 44
Abbildung 17: Klassifikation von Korrespondenzen ... 51
Abbildung 18: Mehr- und Eindeutigkeit von Korrespondenzen ... 51
Abbildung 19: Beispiel zum Beweis ... 53
Abbildung 20: Deckelung der Korrespondenzkandidaten ... 54
Abbildung 21: Korrespondenzen handgeschriebener Ziffern, Quelle: [PtMat98] ... 57
Abbildung 22: Mehrdeutige Registrierung ... 58
Abbildung 23: Falsche Korrespondenzfindung ... 59
Abbildung 24: Eingabefeld für den Targetnamen ... 61
Abbildung 25: Checkbox für „Automatic Name“ ... 61
Abbildung 26: Grünes Licht ... 61
Abbildung 27: Auslöser der Korrespondenzfindung ... 62
Abbildung 28: Fast-Operator, Quelle: [Fast05] ... 72
Abbildung 29: Förstner-Operator ... 73
Abbildung 30: Zelleinteilung von Scans ... 74
Abbildung 31: Muster in Scans ... 75
Abbildung 32: Identische Punkte in Scans ... 76
Abbildung 33: Optische Kreuzungspunkte in Scans ... 76
Abbildung 34: Scheinbare Schattenpunkte in Scans ... 77
Abbildung 35: Skalierung von Masken ... 78
Abbildung 36: Auswahl gefilterter markanter Punkte ... 79

Tabellen

Tabelle 1: Aufbau von Targets ... 15
Tabelle 2: Abstände und Winkel zwischen Targets ... 21
Tabelle 3: Herleitung der tolerierbaren Fehler ... 23
Tabelle 4: Aufbau von Relationen ... 28
Tabelle 5: Relationen als Listeneinträge ... 29
Tabelle 6: Sortierte Relationslisten ... 29
Tabelle 7: Äquivalenzmengen ... 30
Tabelle 8: Initialisierung der Ähnlichkeitswerte ... 31
Tabelle 9: Verbindungen zwischen Matrixeinträgen ... 32
Tabelle 10: Dreifache Neuberechnung der Ähnlichkeitsmatrix ... 33
Tabelle 11: Kandidatenfindung durch Maximumvergleich ... 34
Tabelle 12: Beispiel zur Eliminierungs-Variante ... 38
Tabelle 13: Parameter zufallserzeugter Targetwolken ... 41
Tabelle 14: Parameter für den Standardtest ... 42
Tabelle 15: Parameter für den Paralleltest ... 42
Tabelle 16: Parameter zur Ähnlichkeitsmatrix ... 44
Tabelle 17: Parameter zur Toleranz ... 47
Tabelle 18: Grad der Automatisierung ... 81

Diagramme

Diagramm 1: Variation von RECALC_NUM und LINE_MAX_THRESH ... 45
Diagramm 2: Variation von TOTAL_MAX_THRESH ... 46
Diagramm 3: Variation von PARALLEL_ANGLE_MAX ... 48
Diagramm 4: Künstliches Löschen von Relationen ... 49

UML-Diagramme

UML 1: Aktivität Korrespondenzen finden ... 64
UML 2: Aktivität Relationen berechnen ... 65
UML 3: Aktivität Äquivalenzmengen berechnen ... 65
UML 4: Aktivität Ergebnis prüfen ... 65
UML 5: Aktivität Ähnlichkeitsmatrix erzeugen ... 66
UML 6: Aktivität Ähnlichkeitsmatrix neu berechnen ... 66
UML 7: Aktivität Ähnlichkeitsmatrix Ergebnis berechnen ... 67
UML 8: Klassendiagramme − Targets ... 68
UML 9: Klassendiagramme − Korrespondenzen ... 68

Index

2D-Bilder 9
3D-Laserscanner 9
3D-Primitiven 10
3D-Scan 9
Abbildung 13
Abbildungsverzeichnis 84
Ablauf Korrespondenzfindung 17
Absolutes Maximum 34, 67
Abstand 21, 22
Abstände und Winkel 21
Abstract 7
Abteilung 1
Abweichungen 15, 37
Adjazenzmatrix 56
Adresse 1
Ähnlichkeit von Targets 31, 32
Ähnlichkeitsmatrix 31
Ähnlichkeitsmatrix Ergebnis 67
Ähnlichkeitsmatrix erzeugen 66
Ähnlichkeitsmatrix neu berechnen 66
Aktivieren der CheckBox 62
Algorithmus 27
Alleinstellung 71
Analysen 41
Änderungsaufwand 15
Anforderungen 13
Anfrage 25
ANGLE_ACCURACY_MAX 47
Annahmen zur Komplexität 54
Anwender 81
Anwendungsbereiche 9
Anwendungsfälle 62
Äquivalente Nachbarn 32
Äquivalenzmenge 30
Äquivalenzmengen berechnen 65
Aufbau 11
Aufbau der leeren Matrix 31
Aufbau von Relationen 28
Aufbau von Targets 15
Auflösung 78
Aufwand 54, 81
Ausblick 81
Ausdünnen des Scans 78
Ausgabe 27
Auslöser Korrespondenzfindung 62
Ausprobieren 39
Ausrichtung 81
Ausschließliche Testregistrierung 39
Authority 24
Autokorrelationsmatrix 73
Automatic Name 61
Automatische Registrierung 81
Automatische Targeterkennung 71
Automatisierung 13, 81
Autor 1
Benutzeroberfläche 61
Berechnung Äquiv.mengen 27, 30, 65
Berechnung Relationen 27, 28, 65
Bereich der Registrierung 13
Beschleunigung Neuberechnung 38
Bestimmung Korrespondenzen 27, 34
Bestimmung Korrespondenzkand. 34
Betreuer 1
Beweis zur Korrektheit 52
Bilddaten 13
Bit-Addierer 67
Breitensuche 52
Bresenham-Kreis 72
Checkbox 61
Cluster-Registrierung 63, 69
colMax (Spaltenmaximum) 34
CorrespondenceCheck 69
CorrespondenceSearch 64, 68
CorrespondenceTable 64, 66, 68
CR-Klassifikation 1
Datenelemente 68

Datenfelder 68
Datenfelder eines Objektes 61
Datenstrukturen 17
Deaktivieren der Checkbox 63
Deaktivierung Korr.suche 62
Deckelung 54
Definitionen 51
Diagrammverzeichnis 85
Diskretisierung 72
Distanz 21
Drehung des Scanners 14
Dreieck 19
Dreieckskombinationen 19
Dreiecksrelationen 19
Ebene 10
Ebenentarget 15
Eindeutige Korrespondenz 51
Einfache Fehlertoleranz 15
Einflussgrößen 41
Eingabe 27
Eingabefeld 61
Einleitung 9
Einseitige Relationen 18
Einstellungen des Programms 62
Einteilung Automatisierung 81
Eintragseliminierung 38
Eliminierung von Einträgen 38
Entfernen eines Objektes 63
Entfernen eines Scans 63
Entfernungssprung 76
Ergebnis berechnen 67
Ergebnis prüfen 65
Erkennung eines Objektes 62
Erkennung von Objekten 81
Erklärung 94
Erneute Korrespondenzfindung 39
Erweiterung des Abstandbegriffes 22
Erzeugen der Ähnlichkeitsmatrix 66
Extrembeispiele 58
FAST-Operator 72
Fehlertoleranz 15, 23
Finale Lösung 35
Find Correspondences 62
Finde alle Korrespondenzen 63
Finde Korrespondenzen 62
Förstner-Operator 73
Fotografie 9
Frobeniusnorm 38
Funktionen 68
Gefilterte markante Punkte 79
Gefundene Elemente 25
Gelöschtes Objekt 63
Genauigkeit 13
Geometrische Objekte 61, 71
Geradentarget 15
Gesamtbild 10
Gewichte 37
Gewichtung nach Abweichung 37
Gewöhnliche Registrierung 63
Gierwinkel 14
Gleiche Punkte 76
Gleiche Relationen 30
Gleichverteilung 74
Globale Relationen 20
Globale Selektion 77
Globale Sicht 78
Globale Transformation 13
Grad der Automatisierung 13, 81
Gradientenbild 73
Graph 17, 20
Graph Matching 56
Graphen aus Bildern 56
Graustufenbilder 71
Grünes Licht 61
Grundtyp 15
GUI-Elemente 61
Handgeschriebene Ziffern 57
Herleitung tolerierbarer Fehler 23
Hilfsmittel 94
Hintergrundwissen 81
Hohe Ähnlichkeitswerte 38
Homogenisierung 74

Hub 24
Hubs and Authorities 24
i (Metrik) 43
Identische Punkte 76
Identische Targets 10
Implementierung 61
InclinometerTarget 68
Index 87
Informationsverlust 9
Inhaltsverzeichnis 3
Initiale Ähnlichkeitsmatrix 27, 31
Initialer Ähnlichkeitswert 31
Inklinometer 14
Inkonsistente Korrespondenz 51
Inkorrekte Korrespondenz 51
Inkrementelle Neufindung 39
Institut 1
Intention 9
Interessen-Wert 77
Interest-Operatoren 71
Internet-Suchmaschine 25
Invariante Eigenschaften 17, 18
Iterierte Neuberechnung 27, 33
JITTER_MAX 41
k (Metrik) 43
Kandidaten 34, 67
Kandidatenkombination 67
Kandidatenwahl 35
Kanten 17, 20
Kartesische Koordinaten 14
Keine Testregistrierung 39
Klasse CorrespondenceSearch 64
Klasse CorrespondenceTable 66
Klassendiagramme 68
Klassenübersicht 68
Klassifikation 13
Klassifikation Korrespondenzen 51
Klassische Berechnung 21
Knoten 17
Kombination der Operatoren 79
Kompass 14
Komplexität 54
Konfiguration 41
Konsistente Korrespondenz 51
Kontext 81
Kontextmenü Scan 62
Konzept 17
Koordinatensystem 10
Korrekte Korrespondenz 51
Korrektheit 52
Korrespondenzbestimmung 27
Korrespondenzen 34
Korrespondenzfindung 10, 64, 81
Korrespondenzkand.-Deckelung 54
Korrespondenzkandidaten 34, 67
Korrespondenzprüfung 69
Korrespondenztypen 51
Korrespondierende Relationen 30
Kreis 15
Kreislinie 72
Kreisradius 30
Kreuzungspunkte 75, 76
Kriterien zur Kandidatenwahl 35
Künstliche Targets 81
Kugel 10, 15
Kugelradius 30
Lage 15
Lageabhängige Eigenschaften 18
Lageunabhängige Eigenschaften 18
Laserscanner 9
Laserstrahl 9, 71
Leere Matrix 31
Lichtsymbol 61
Light 61
LINE_MAX_THRESH 34, 44
Lineare Hülle 20
Link 68
Literatur 83
Lokale Sicht 78
Lokalmenge 27
Lokalwolke 14
Löschen eines Objektes 63

Löschen eines Scans 63
Lösungsansätze 17
Luftfahrnorm 14
Magnetische Störfelder 14
Manuelle Registrierung 81
Manueller Aufwand 81
Markante Punkte 71
Marker im Bild 62
Markierter Bereich 62
Maske 73
Maskenausschnitt 78
Maskenskalierung 78
Maskenstreckung 78
Match Matrix 56
Matching Algorithmen 56
Matrix Neuberechnung 33
Matrixeintrag 31
Matrixmaximum 67
Matrixnormierung 38
Maximale Korrespondenz 51
Maximaler Fehler 23
Maximaler Rotationsfehler 15
Maximaler Teilgraph 52
Maximaler Translationsfehler 15
Mehrdeutige Korrespondenz 51
Mehrdeutige Lösungen 58
Mehrseitige Relationen 20
Messgenauigkeit 15
Metriken 25, 43
Modell 9
Motiv 9
Muster 17, 71, 75
Musterpunkte 75
Nachbartarget 32
Name des Objekts 61
Natürliche Targets 15, 81
Negative Metrik 25
Neuberechnung 27, 33
Neuberechnung Ähnlichkeitsmatrix 66
Nickwinkel 14
Normalenvektor 15
Normierung 38
Oberflächenelemente 61
Objekteigenschaften 61
Objekterkennung 62, 81
Objektname 61
Operatoren 71
Optimierung FAST-Operator 72
Optimierungen 74
Optische Kreuzungspunkte 76
Orientierung 13
P_ORTHOGONAL 42
P_PARALLEL 42
P_TARGETS_ADD 42
P_TARGETS_CLOUD 42
P_TARGETS_REM 42
P_WEIGHT_PIPES 42
P_WEIGHT_PLANES 42
P_WEIGHT_POINTS 42
Paarweise Registrierung 63
PageRank 24
PARALLEL_ANGLE_MAX 47, 48
PARALLEL_DIST_DIFF 47
PARALLEL_DIST_MAX 48
PARALLEL_TESTS 42
Parallelabstand 22, 30
Paralleltest 41
Parameter der Transformation 14
Parameter für Targetwolken 41
Parameter zur Ähnlichkeitsmatrix 44
Parameter zur Toleranz 47
Parameterkonfiguration 43
Perspektive 79
Pin im Bild 62
PipeTarget 68
PlaneTarget 68
Platzaufwand 54
Platzkomplexität 54
Point Matching 57
Point of Interest 71
PointTarget 68
Polygone 20

POSITION_ACCURACY_MAX 47
POSITION_MAX 41
Positionierung 81
Positionsfehler 13
Positionsgenauigkeit 67
Positionsvektor 15
Positive Metrik 25
Precision 25
Precision and Recall 25
Primitiven 10
Prüfen des Ergebnisses 65
Prüfer 1
Prüfung der Korrespondenzen 69
Punkt 10
Punktmuster 71
Punkttarget 15
Punktumgebung 71
Punktwolke 9
Radius 30
Rahmenbedingungen 13
RANDOM_TESTS 42
Raster 72
Raumecken 71, 78
Raumwinkel 9
Realität 9, 81
RECALC_NUM 44
Recall 25
Referenzmenge 27
Referenzwolke 14
Reflektion des Laserstrahls 71
Regelmäßiges Muster 71, 75
RegistrationTarget 68
Registrierung 10, 35, 81
Reichweite 41
Relationen 18, 28, 68
Relationen berechnen 65
Relative Lage 18
Relevante Elemente 25
Resultierende Fehlertoleranz 23
Richtungsvektor 15
Robustheit 49, 79
Rollwinkel 14
Rotation 57
ROTATION_ERROR_MAX 47
Rotationsfehler 15
Rotierender Laserstrahl 9
rowMax (Zeilenmaximum) 34
Rückmeldung 61
Satz zur Korrektheit 52
Scan 9
Scandaten 81
Scanner 9
Scannerlage 14
Scannerposition 14
Scansoftware 81
Schachbrettmuster 15
Schattenlinien 77
Scherung 57
Schwerkraft 14
Schwerpunkt 20, 22
Selektion von Objekten 81
Seltenheit 71
Setzen der Checkbox 62
Setzen der Verbindungen 32
Setzen initialer Ähnlichkeitwerte 31
sim (Ähnlichkeitswert) 34
Skalar 20, 28
Skalare Eigenschaften 21
Skalierung 57
Skalierung der Masken 78
SlabTarget 68
Software 81
Spaltenkandidat 34
Spaltenmaximum 34, 67
Speicherplatz 54
Spezielle Targets 15
Sprung der Entfernung 76
Stabilität 71
Standardabweichung 67
Standardtest 41
Standort 10
Starre Transformation 13

Streckung der Maske 78
Strukturbaum 62
Studiengang 1
Suchbaum 52
Suche nach markanten Punkten 71
Suchmaschine 25
Summary 7
Symmetrie 58
Symmetriebrecher 79
Tabellen 85
Tabellenmaximum 67
Tabellenverzeichnis 85
TableEntry 68
Targeterkennung 71
Targetmenge 1
Targets 10, 15, 81
TARGETS_ADD 42
TARGETS_CLOUD 42
TARGETS_REM 42
Teilgraph 52
Test der Korrespondenzen 69
Test zur Robustheit 49
Testregistrierung 37, 39, 67
Theorie 51
Tiefeninformation 71, 79
Titel 1
Toleranz 15
TOTAL_MAX_THRESH 34, 44
Totales Maximum 34, 67
totalMax (Tabellenmaximum) 34
Transformation 13, 14
Translation 57
TRANSLATION_ERROR_MAX 47
Translationsfehler 15
Typ 30
Überdeckung 13
Überlappung 13, 52
UI-Elemente 61
Umbenennen eines Objektes 63
Umgebung 71
UML-Diagrammverzeichnis 85
Unscharfe Kanten 78
Untertyp 15, 30
Varianten 37, 49
Variation 44, 45
Verbindungen 32
Verbindungsvektor 20
Verkürzungsmerkmal 9
Verweise 31
Verzeichnis 84
Vollautomatische Registrierung 71
Voraussetzung Korr.findung 71
Voraussetzungen zur Komplexität 54
Vorgehensweise 11
Wahl der Kandidaten 35
Wand 15
Wanddicke 30
Webseite 24
WEIGHT_PIPES 42
WEIGHT_PLANES 42
WEIGHT_POINTS 42
Wiedererkennung 71
Wiederkehrende Muster 59
Winkel 21
Winkelgenauigkeit 67
Zeilenkandidat 34
Zeilenmaximum 34, 67
Zeitaufwand 55
Zeitkomplexität 55
Ziel der Registrierung 14
Zufallsgenerierte Graphen 56
Zusammenfassung 7
Zusammenführen der Punktdaten 10
Zweidimensionale Bilder 9
Zweiseitige Relationen 20
Zylinder 10, 15

Erklärung

Ich versichere, dass ich diese Arbeit selbstständig verfasst und nur die angegebenen Hilfsmittel verwendet habe.

_____

Ivan Bogicevic